United States Patent [19]

McKee et al.

[11] **Patent Number:** **5,477,531**

[45] **Date of Patent:** **Dec. 19, 1995**

[54] **METHOD AND APPARATUS FOR TESTING A PACKET-BASED NETWORK**

[75] Inventors: **Neil H. McKee**, Horfield; **Peter Phaal**, Bradley Stoke, both of England

[73] Assignee: **Hewlett-Packard Company**, Palo Alto, Calif.

[21] Appl. No.: **162,114**

[22] PCT Filed: **May 28, 1992**

[86] PCT No.: **PCT/GB92/00967**

§ 371 Date: **Dec. 13, 1993**

§ 102(e) Date: **Dec. 13, 1993**

[87] PCT Pub. No.: **WO92/22967**

PCT Pub. Date: **Dec. 23, 1992**

[30] **Foreign Application Priority Data**

Jun. 12, 1991 [EP] European Pat. Off. 91305315

[51] Int. Cl.⁶ H04J 3/26; H04L 1/14; H04L 12/26

[52] U.S. Cl. 370/17; 370/60; 370/94.1

[58] Field of Search 370/13, 17, 60, 370/60.1, 94.1, 94.2; 371/20.1, 20.2

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,551,833 11/1985 Turner .
5,101,402 3/1992 Chui et al. 370/17
5,343,465 8/1994 Khalil 370/17
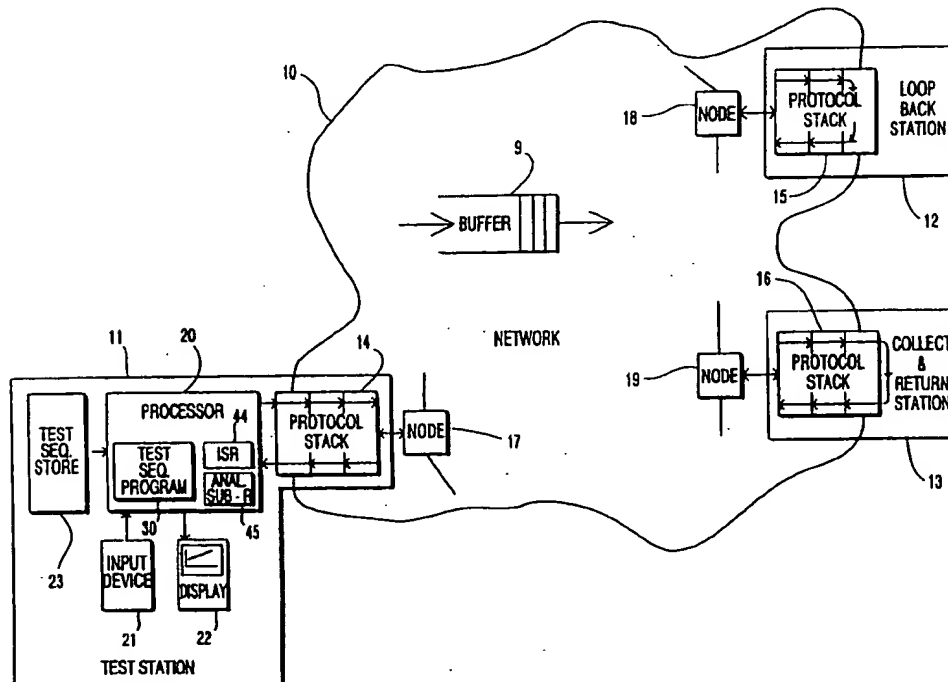
OTHER PUBLICATIONS

"Measuring a Local Network's Performance", by P. D. Amer et al, Data Communications, vol. 12, No. 4, Apr. 1983, pp. 173–182.
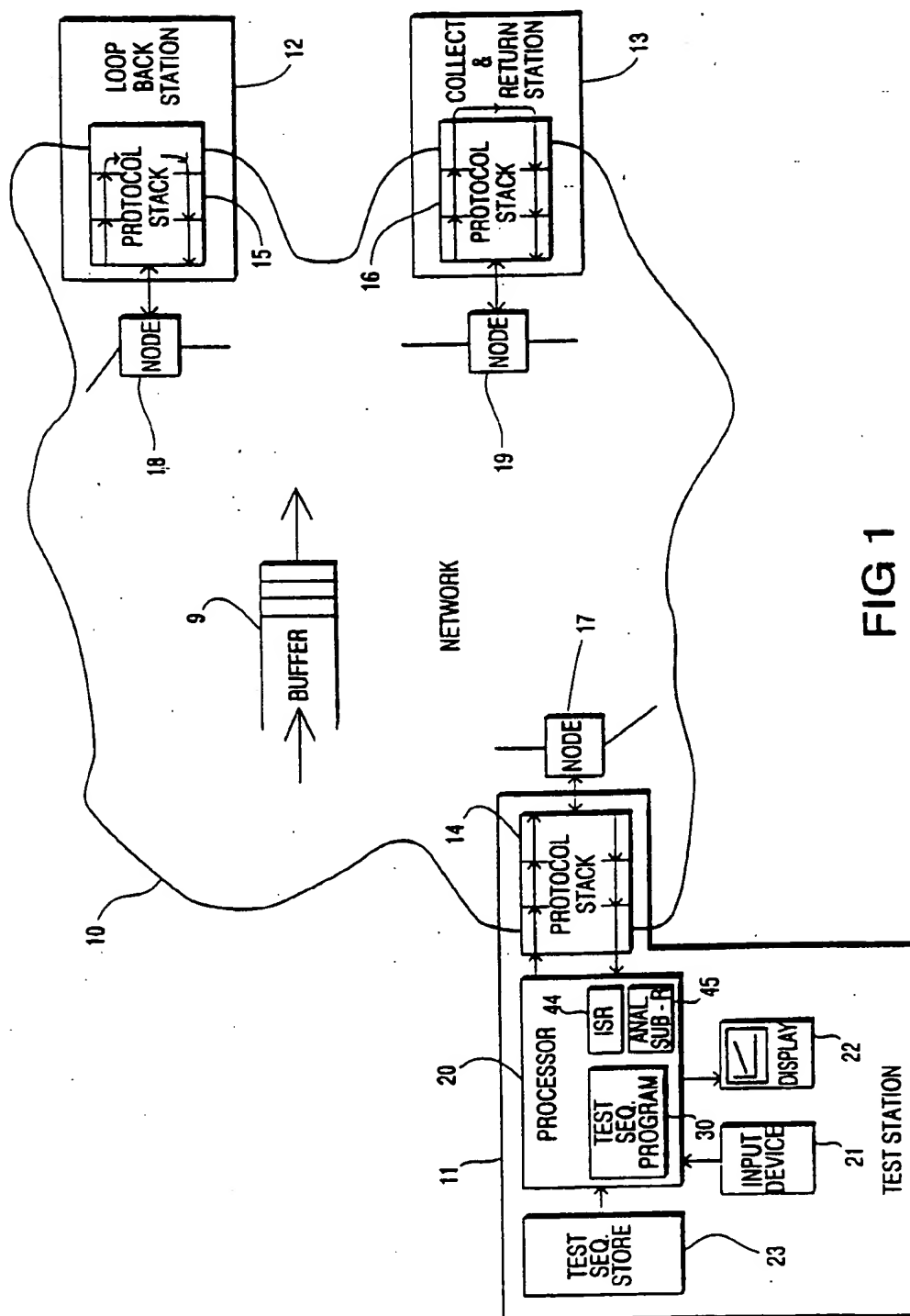
"Statistical Monitors for Local Area Networks", D. C. Feldmeier, 11th Conference on Local Computer Networks, Oct. 8, 1986, pp. 142–146.
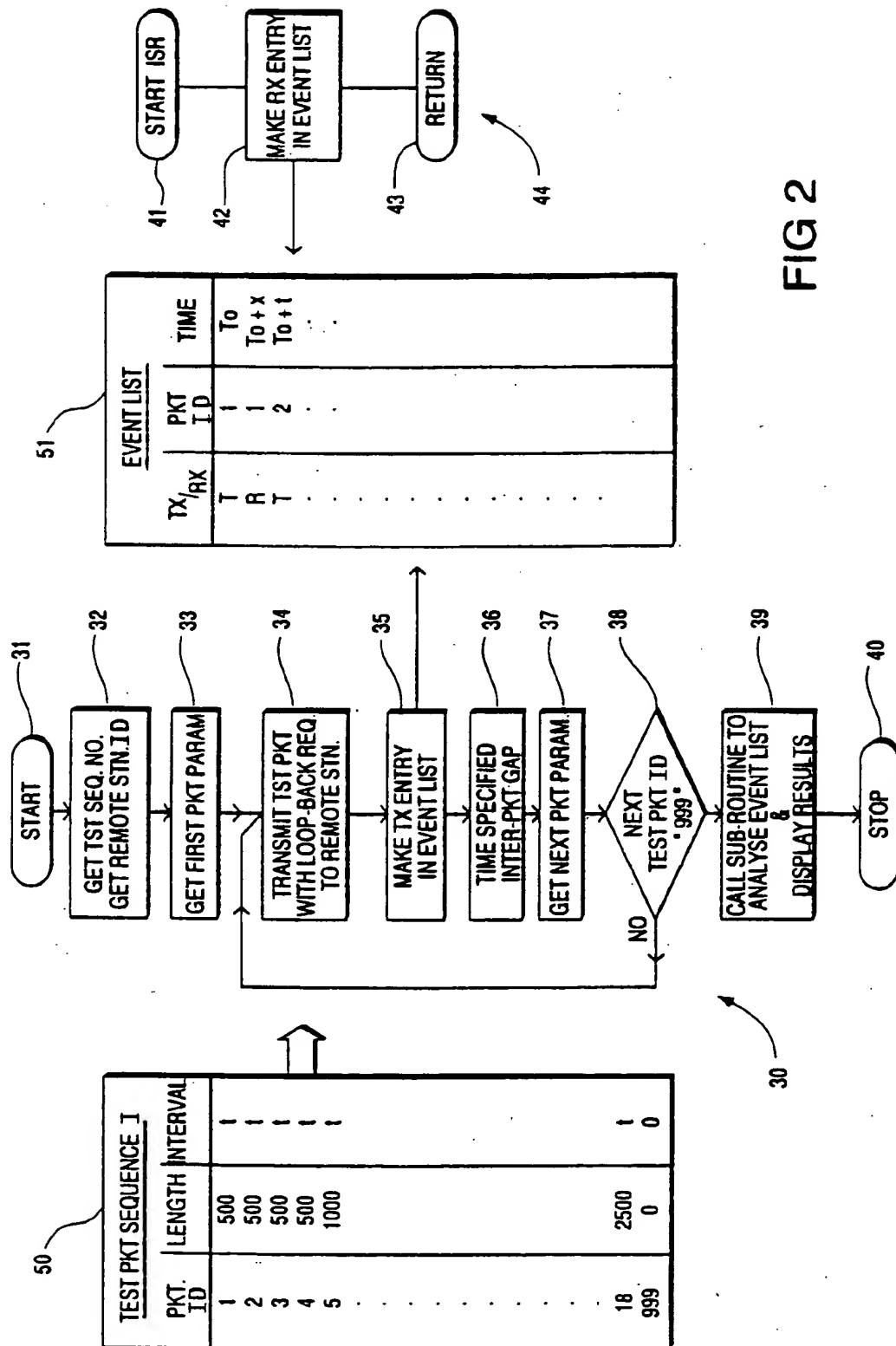
"Internetworking With TCP/IP", by Douglas Comer, 2nd edition 1991, Prentice–Hall International, Inc.

Primary Examiner—Wellington Chin

[57] **ABSTRACT**

In packet-based networks (10), it is often desired to test communications between two specific stations (11, 12). This can generally be effected from a first one of the stations (11) by requesting the other station (12) to 'loop-back' a test packet sent from the first station. The first station (11), on receiving back the test packet, can thereby ascertain that not only is communication with the other station (12) possible, but it can also measure the round trip time. However, more complex characteristics of the transmission path between the stations (11, 12) are not ascertainable in this manner. The transmission of a predetermined sequence of packets permits such characteristics to be determined by observing the effect of the network (10) on the sequence considered as a whole. Thus by varying packet size in a sequence, characteristics such as bandwidth, propagation delay, queuing delay and the network's internal packet size can be derived. The use of bursts of packets enables buffer size and re-sequencing characteristics to be determined.

14 Claims, 6 Drawing Sheets

FIG 1

FIG 2

FIG 3A

FIG 3B

$$\text{BANDWIDTH} = \frac{1}{\text{SLOPE}}$$

MIN.
DELAY

PROPAGATION DELAY

70

500   1000   1500   2000   2500

PKT. SIZE

FIG 4

72

71

MEAN
DELAY

AV. QUEUE DELAY

70

500   1000   1500   2000   2500

PKT. SIZE

FIG 5

FIG 6

FIG 7

PKT.
SIZE

2000
1500
1000
500

T

TRANSMIT TIME

FIG 8

%

RXD

TX3	TX3	
TX2	TX2	TX2
TX1	TX1	TX1

1   2   3

RXD POSITION IN BURST
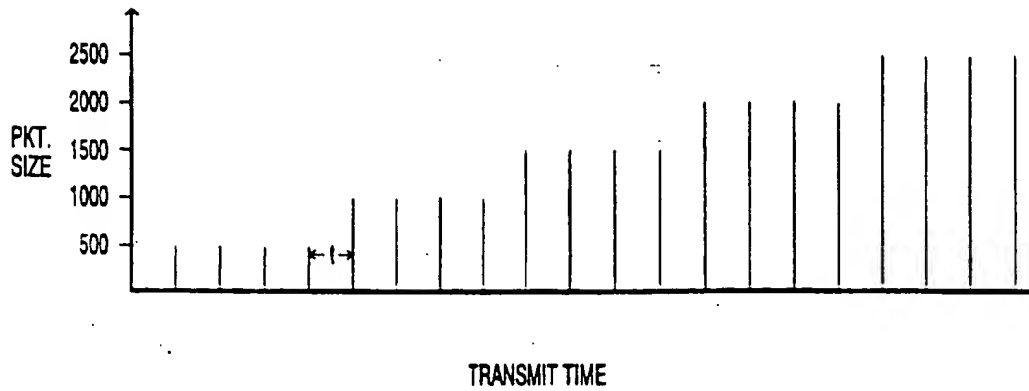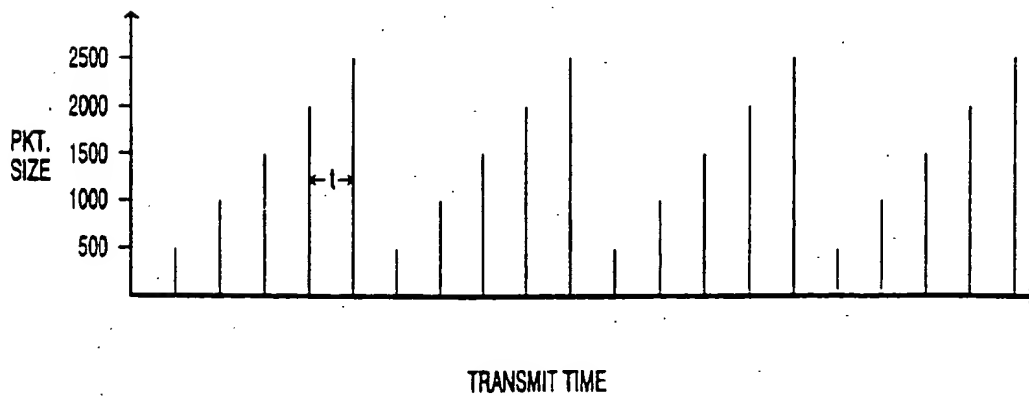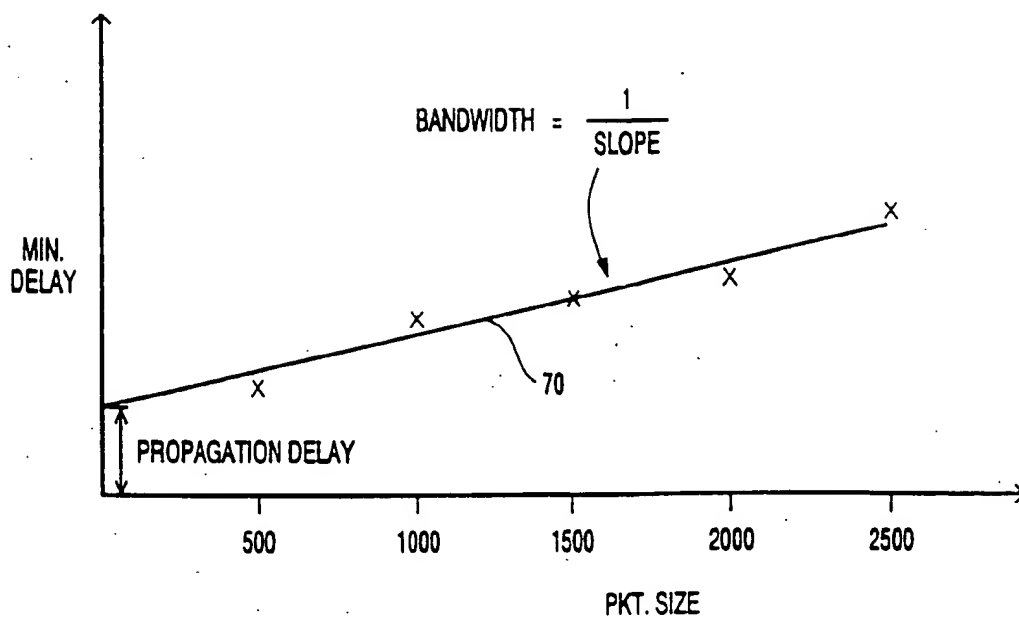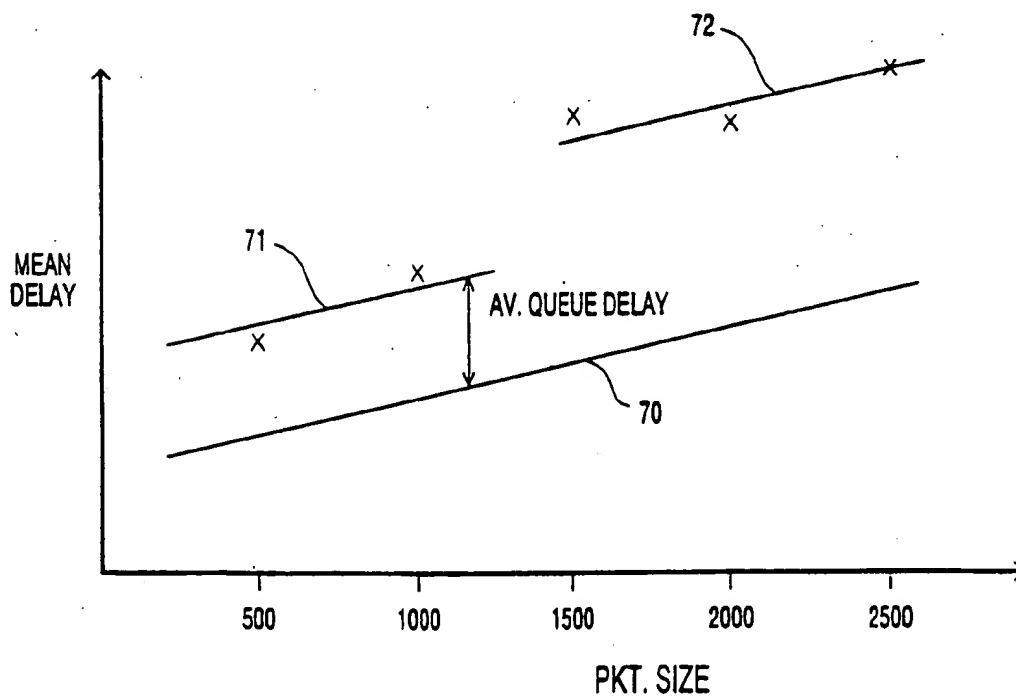
FIG 9

METHOD AND APPARATUS FOR TESTING A PACKET-BASED NETWORK

TECHNICAL FIELD

The present invention relates to a method and apparatus for testing a packet-based communications network such as a computer network.

BACKGROUND

In packet-based networks, it is often desired to test communications between two specific nodes on the network. This can gener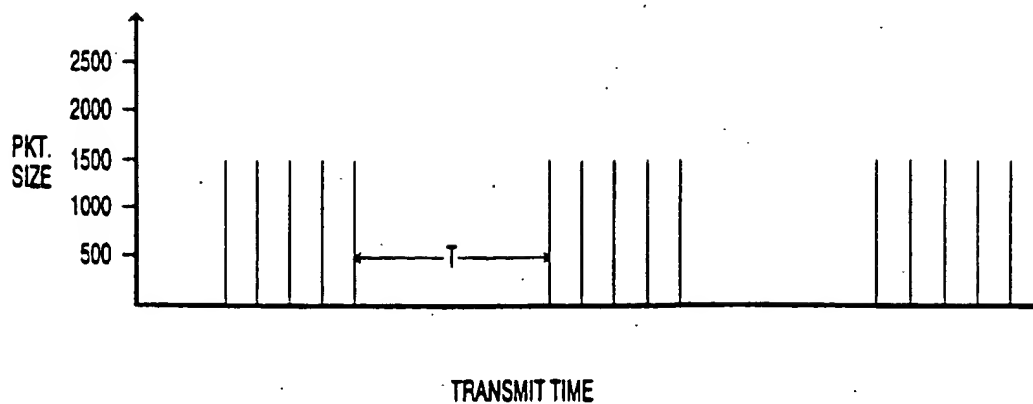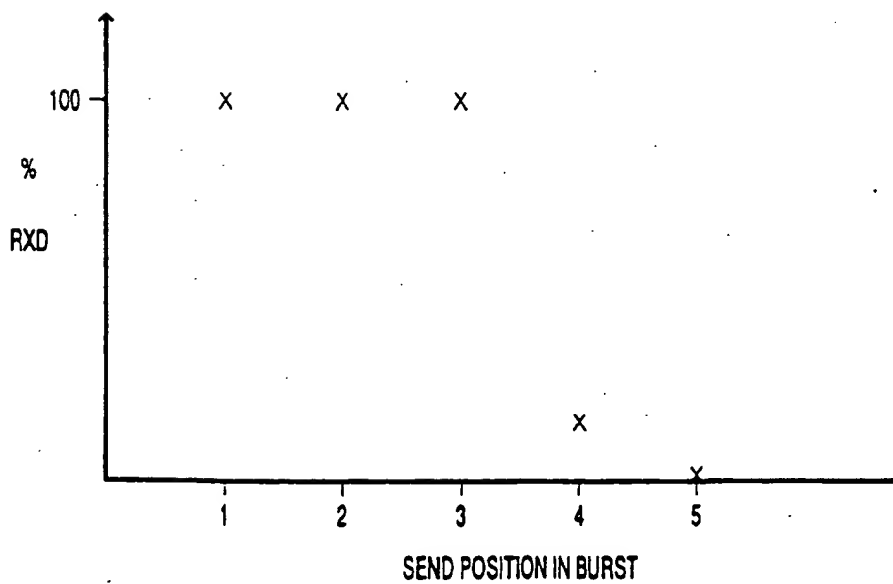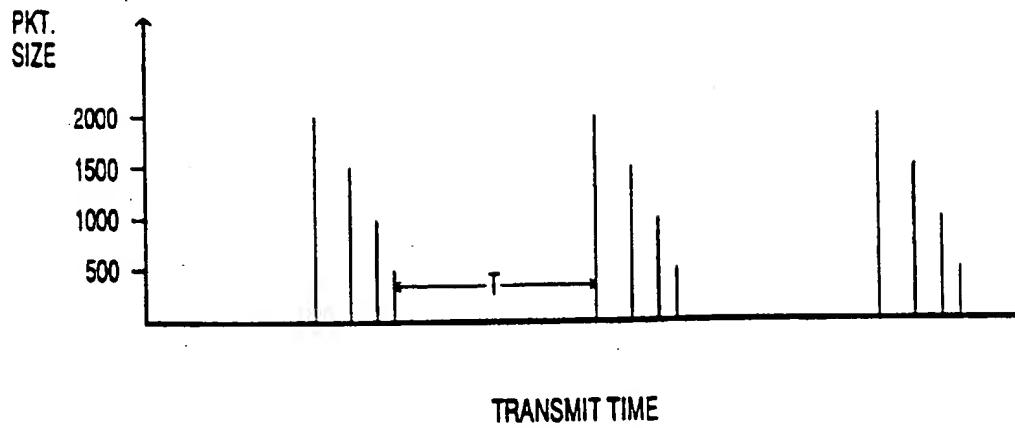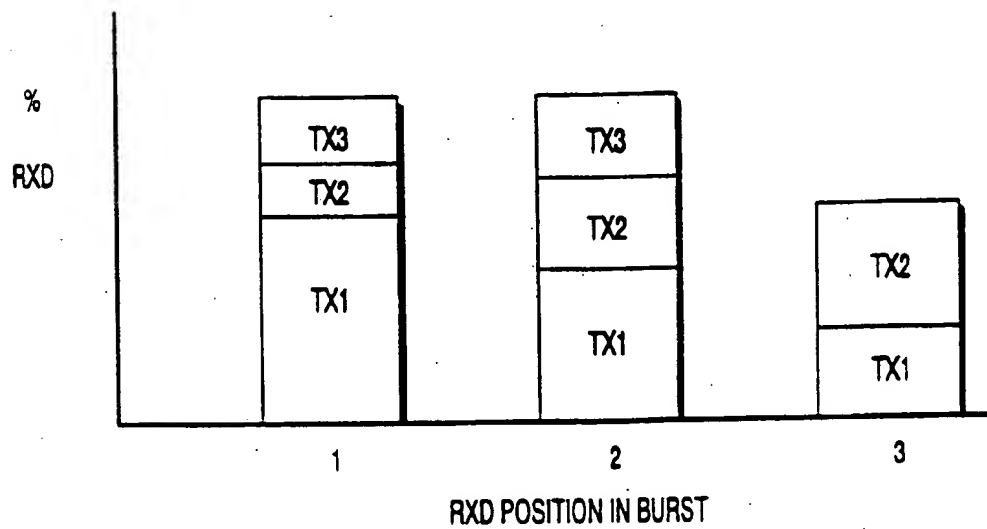ally be effected from a first one of the nodes by requesting the other node to 'loop-back' a test packet sent from the first node. The first node, on receiving back the test packet, can thereby ascertain not only that communication is possible with the other node, but also the round trip time for the packet. Typically such a test would be performed by an instrument such as the HP 4982A LAN protocol analyzer (available from Hewlett-Packard Company, Palo Alto, Calif.) connected to the network at the first node.

Inter-networking protocols such as the ARPA Internet Protocol (IP) may also provide a facility for determining reachability and round trip time by use of a looped-back test packet. Thus, in IP networks the Internet Control Message Protocol CiCMP) allows control and information messages, including echo request and echo reply messages, to be passed in the data portion of IP datagrams between IP software on different hosts and gateways. Other messages allowed by ICMP include timestamp request and timestamp reply messages which permit a transit time estimate to be made in both transit directions. Certain computer operating systems such as the Hewlett-Packard HPUX operating system permit users to send ICMP echo requests using a command named "ping" with the users being able to specify the number and size of test packets for which round trip times are returned.

It may also be noted that in the ARPA Transmission Control Protocol (TCP), a similar measurement is continually made as part of the transmission control process. More particularly, the round-trip time is measured between the transmission of a packet and receipt back of an acknowledgement from the destination node; this round-trip time is averaged continually into a smoothed round-trip time estimate which is then used to control the retransmissions time-out parameter RTO.

The use of looped-back test packets has thus been restricted to determining the directly-observable round-trip time characteristic of the transmission path between two nodes.

It is an object of the present invention to permit further network characteristics to be determined from the operation of transmitting packets between two nodes, these characteristics being those which are not discernible from the transmission of a single packet between the nodes.

DISCLOSURE OF THE INVENTION

According to one aspect of the present invention, there is provided a method of testing a packet-based network to ascertain characteristics of packet transmission between first and second nodes on the network, said method comprising the steps of transmitting packets between said nodes, receiving the packets at one said node, and correlating packet transmission and reception to derive correlation data indicative of a said transmission characteristic, characterised in

that said packets are transmitted as a sequence in which the packets are in a predetermined relationship to each other, said correlation data being so derived as to be sensitive to this relationship whereby to enable a characteristic of transmission to be determined that is unobservable from the passage of a single packet.

Preferably, at least one of the sequence parameters comprising inter-packet spacing and packet size, varies through said sequence in accordance with said predetermined relationship.

In one test according to the invention, said sequence of packets comprises for each of a multiplicity of packet sizes, a respective plurality of packets, the packets of said sequence being transmitted in isolation from each other.

The correlation data for this test will comprise the minimum and/or the mean packet journey time for each packet size. Where the minimum journey times are derived, these are preferably graphically displayed against packet size; the slope of a straight line placed through the points on this graphical display will give an indication of the network bandwidth whilst the intercept of the same straight line on the time axis will provide an indication of propagation delay. Of course, the minimum journey times could be processed computationally in order to derive the same network characteristics. Where the correlation data includes both the minimum and the mean journey times, graphical display of both against packet size enables the mean queuing time for packets transmitted between the nodes to be derived, this queuing time being the difference between the minimum and mean times for any particular packet size; again, the queuing time could be computed rather than graphically derived. Where the correlation data comprises the mean journey times, graphical display of these times against packet size can provide an indication of the network's internal packet size since if the network splits up an original sequence packet into smaller packets, this will be reflected in an increased journey time which should show up as a step function on the graphical display. Again, the networks internal packet size could be derived by computation from the correlation data.

In another test according to the invention, the sequence of packets transmitted from the first node comprises a succession of isolated packet bursts each made up of a plurality of packets transmitted immediately one after another, the correlation data being so derived as to indicate for each packet position within said bursts as transmitted, the average packet loss rate. If the minimum buffer size in the transmission path between the nodes under consideration is less than the number of packets included in each burst, the loss rate of packets should increase significantly as the buffer size is exceeded by packets occurring later in each burst. This test therefore enables the minimum internal buffer size to be determined.

In a further test according to the invention the sequence of packets transmitted comprises at least one isolated packet burst made up of a plurality of packets that are transmitted immediately one after another and are of decreasing size through the packet; in this case, the correlation dam is derived so as to indicate any difference in packet sequencing between the transmitted and received bursts. Such re-sequencing may be present where the network includes devices that prioritise short packets.

BRIEF DESCRIPTION OF THE DRAWINGS

Three network tests according to the invention, and test apparatus embodying the invention, will now be particularly

described, by way of non-limiting example, with reference to the accompanying diagrammatic drawings, in which:

FIG. 1 is a diagram illustrating use of the test apparatus to ascertain characteristics of a network between two given network nodes;

FIG. 2 is a diagram illustrating a main test sequence program of the test apparatus and associated data structures.

FIG. 3A illustrated a first test sequence of packets for use in carrying out a first one of the network tests;

FIG. 3B illustrates a second test sequence of packets for use in the first network test;

FIG. 4 is a graph of minimum packet delay against packet size, derived by the first network test;

FIG. 5 is a graph of mean packet delay against packet size, derived by the first network test;

FIG. 6 illustrates a test sequence of packet bursts for use in a second one of the network tests;

FIG. 7 is a graph derived by the second network test and from which the minimum network path buffer size can be determined;

FIG. 8 illustrates a test sequence of packet bursts for use in a third one of the network tests; and

FIG. 9 is a graph derived by the third network test and from which any packet re-sequencing can be identified.

BEST MODE FOR CARRYING OUT THE INVENTION

FIG. 1 illustrates a packet-based communication network 10 to which three stations, 11, 12 and 13 are connected at respective network nodes 17, 18 and 19.

The network 10 will generally be made up of one or more links operating to the same or different protocols and will typically include a number of queuing devices such as bridges and gateways, generally depicted by the buffer 9 in the FIG. 1 network. These devices serve to queue packets received at the devices pending appropriate processing of the packets (for example routing to the appropriate network link). The different links of the network will typically operate at different speeds with different internal packet sizes.

The stations 11, 12 and 13 interact with the network 10 through respective protocol stacks 14, 15 and 16, which will generally be provided by communications software running on a processor of the associated station (this processor may be either a dedicated communications processor or a central processor of the station, running in a multi-tasking mode). The protocol stacks may be of any form suitable for the particular links to which the corresponding stations are connected. Implementations of such protocol stacks are well known in the art and will therefore not be described in detail; however, these stacks may include, by way of example, the TCP/IP protocol suite, that is, the Transmission Control Protocol/Internet Protocol suite (for a further description of TCP/IP see, for example, "Internetworking with TCP/IP", by Douglas E. Comer, second edition 1991, Prentice-Hall International, Inc). The lower protocol levels may be in accordance with the IEEE 802.2 link level protocol and appropriate ones of the IEEE 802.3/4/5/6 physical layer protocol standards. Although only three protocol layers have been illustrated for each stack, 14, 15 and 16 in FIG. 1, this is purely done to simplify illustration and more or less protocol layers may be present. As will become clear below, in relation to the network characteristic measurements to be

described, at least part of each protocol stack is within the network 10.

Station 11 is a test station incorporating the test apparatus embodying the present invention. The test station 11 is operative to assess the characteristics of the transmission path between the test station and a remote station (hereinafter, the target station) by sending out sequences of test packets to the target station. In the preferred implementation of the invention, the target station is arranged to loop back the test packets to the station 11; in FIG. 1, station 12 constitutes such a target station effecting test packet loopback (for example, stations 11 and 12 may both operate the IP protocol and the test packets transmitted by the station 11 may either be echo request or timestamp request messages under the ICMP protocol to which station 12 will respond accordingly). The station 11 on receiving back the test packet sequence, correlates the transmission and reception of the test packets in order to derive characteristics of data transmission across the network 10 between the test and target stations 11 and 12. Station 11 is, for example, constituted by a portable instrument which may be moved between different nodes on the network to measure transmission characteristics between various nodes.

Station 11 comprises a program-controlled processor 20 operative to run a main test sequence program 30, an input device 21 such as a keyboard, a graphical output device 22 and a test sequence store 23 containing sequence specifications for a number of different test packet sequences. Each test sequence specification specifies sequence parameters such as packet size, packet identity, inter packet gap and packet ordering. In response to user input through the device 21, the processor 20 is operative to run the test sequence program 30 to transmit a test packet sequence corresponding to a selected one of the specifications held in store 23, to a user specified target station for loop back to the test station 11. In transmitting packets to the target station, the test sequence program 30 utilizes the services provided by the protocol stack 14 to send and receive packets over the network 10. The basic service provided by the protocol stack 14 is to transmit a test packet of specified length and including specified packet D to a specified remote station with a loop back request indication in the packet header. Upon the packet being returned to the protocol stack 14, an interrupt is generated to the processor 20.

The main test sequence program is operative to carry out any required timing of packet transmission and receipt, or if such a facility is provided by the protocol stack (as with timestamp request messages of the ICMP protocol), to arrange for the stack to provide appropriate timing. In addition, the test sequence program will correlate data on the transmitted and received packets, analyze this data, and output the analysis results to the user via the display 22.

The test sequence program 30 will now be considered in more detail with reference to FIG. 2. As noted above, the program 30 is intended to send out a sequence of test packets to a specified remote station for loop back with the parameters of the test sequence being in accordance with a selected one of a plurality of test packet sequence specifications held in the store 23. Each test packet sequence specification can be represented as a list 50 containing for each of a series of test packets, a packet ID number, the length of the packet in bytes and the interval that is to follow the packet before the next packet in the sequence is transmitted. The list of test packets is terminated by an entry giving a packet ID of "999".

In addition to the test packet sequence list 50, the test

sequence program also makes use of a data structure in the form of an event list 51. This event list lists in time order the transmission and reception of test packets by the station 11. For each event the list contains an indication of whether or not the event concerned is the transmission or reception of a packet CrX/RX in list 51 of FIG. 2), an indication of the ID number of the packet concerned, and the time of the relevant event (packet transmission or reception). Transmission events are entered in the event list 51 by the main test sequence program 30, whilst reception events are entered into the list 51 by an interrupt service routine 44, initiated by the interrupt produced by the protocol stack 14 whenever a packet is received.

The general operation of the test sequence program 30 will now be described:

Upon initiation of the test sequence program (block 31) the user is asked to enter the identity of the test sequence it is wished to run and also to identify the remote station to which the test packets are to be sent for loop back (block 32). Next, the test sequence program 30 retrieves from the relevant test packet sequence fist 50 the parameters of the first test packet of that sequence (block 33). Thereafter the program 30 controls the transmission of a test packet to the specified remote station using the protocol stack 14. The transmitted test packet will include a packet ID number of 1 and be of a length specified in table 50 (block 34). As soon as the protocol stack 14 has been instructed to transmit the test packet, an entry is made in the event list 51 recording the fact that packet number 1 has been transmitted at a particular time (block 35).

Next, the program 30 times an interval corresponding to that set out in the list 50 for packet just transmitted, this interval being the desired delay before the next packet is sent (block 36). At the end of this time out, the parameters of the next packet to be transmitted are fetched from the list 50 (block 37); except in the case where the ID number of this next packet is "999", the program now loops back to block 34 to send this next packet.

The loop constituted by blocks 34 to 38 is thereafter repeated until all the test packets of the current test sequence have been sent. In due course, the end-of-sequence packet with its ID number set to "999" is encountered and at this point the program passes from block 38 to block 39. Block 39 concerns the analysis of the event list and the display of results to the user. Since the process of analysis and display of results will be dependent on the test packet sequence concerned, a separate analysis and display routine 45 will be called by program 30 in dependence on the test sequence concerned. After the analysis and display sub-routine has terminated, the main test sequence program 30 itself terminates (block 40).

Execution of the main test sequence program 30 is interrupted each time a packet is received by the protocol stack. When this occurs the interrupt service routine 44 is run (block 41) and simply involves an entry being made in the event list to record the receipt back of a test packet, the entry identifying the test packet concerned and its time of receipt (block 42). Thereafter the interrupt service routine is terminated (block 43) and the control is returned to the main program.

The time stamping of packet transmission and reception as recorded in the event list is effected from a common clock run by the processor 20, this being true whether the time stamping is effected externally of the protocol stack 14 or as part of the services provided by the stack (in the latter case, the transmit timestamp of a test packet may be entered in the

event list at the same time as the receipt timestamp, rather than earlier, as both will generally be available together from the stack on receipt back of the test packet).

The timing of the inter-packet gap, in block 36 of the main program generally does not require significant accuracy, so that this timing can be done in software with any extension of the interval due to the running of the interrupt service routine 44, being small enough to be ignored. As will become clear below, the main purpose of timing an inter-packet gap is that in certain cases adjacent packets are required to influence the network independently of each other and therefore need to be isolated by a sufficient time interval.

Influence of Network on Packet

When a test packet is sent out over the network to a particular remote station and looped back to the transmitting station, its round trip journey time will depend on a number of factors. More particularly, the round trip journey time will depend on packet length because timing is normally effected between start of packet transmission and conclusion of packet reception, so that the longer a packet is, the greater its round trip journey time. Furthermore, the journey time will also depend on the protocol stack level at which the timing takes place—if timing is effected at one of the lower protocol stack levels, then the journey time will be less than if the timing is effected at a higher level. Similarly, the round trip journey time will be affected by the level in the protocol stack at which loop back is effected at the remote station; generally it is possible to effect loop back both at the link level layer and at the transport layer or higher and, again, the higher the loop back layer, the greater the journey time. It is for this reason that in FIG. 1, the protocol stacks 14, 15 and 16 have been shown as at least partially within the boundary of the network 10. Another factor affecting the round-trip journey time is the speed of each link taken by the packet. A still further factor is the amount of queuing experienced by the packet as it passes through queuing devices such as bridges and gateways. The internal packet size used by the links in the network may also effect round-trip journey time, because if a packet as sent must be split up into several smaller packets for transmission over one or more of the links before being reassembled, then there is a greater likelihood that the greater number of reduced-sized packets will be subject to some delay. Another factor that may influence journey time is the length of the packet as compared to other packets on the network because certain devices on the network may be arranged to give priority to short packets and this could significantly influence the round-trip journey time of a test packet. A test packet may also be subject to other influences such as being lost or being duplicated.

Whilst the basic round-trip journey time of a test packet can be directly measured, the underlying network parameters that may influence journey time cannot be detected from the passage of single test packet across the network and it is for this reason that the present apparatus is arranged to send out sequences of test packets.

First Test Method

Turning now to a consideration in detail of a first test carried out by the apparatus of the present invention, reference is made to FIGS. 3A and 3B which illustrate two possible test packet sequences for use in this test. More particularly, in FIG. 3A a first test sequence is illustrated in

which twenty test packets are sent comprising four packets for each of five different sizes ranging from 500 bytes to 2,500 bytes. Each test packet is separated from its neighbours by a time interval "τ". The time interval "τ" is chosen such that each packet is isolated from its neighbours in the sense that any effect on the network caused by one packet has disappeared by the time the next packet is transmitted.

The test sequence shown in FIG. 3B is similar to that shown in FIG. 3A and includes the same test packets but arranged somewhat differently into four sets of five differently sized packets. However, again the packets are each isolated from each other by a time interval "τ".

Either of the test packet sequences shown in FIGS. 3A and 3B may be used in carrying out the first test. The test packet sequence list 50 illustrated in FIG. 2 shows the list entries for the first few packets of the FIG. 3A sequence.

To carry out the first test, the test sequence program 30 is run and the user chooses either the test sequence of FIG. 3A or of FIG. 3B and identifies the remote station concerned. Thereafter, the program 30 transmits the test sequence and, together with the interrupt service routine 44, builds up the event list 51 indicating the time of transmission and receipt of the various test packets of the selected test sequence. Once all the packets have been transmitted, the test program 30 calls the appropriate analysis and display sub-routine. For the first test, this sub-routine is operative to analyze the event list and provide a further list of packet identity number against round-trip time, this information being readily derived by subtracting the time of transmission of a packet from its time of receipt, as recorded in the event list. Then for each packet size involved in the test packet sequence, the sub-routine derives a minimum and mean round trip time and these values are graphically displayed against packet size on the output device 22.

FIG. 4 shows the graph of minimum delay against packet size for typical test results. As can be seen a straight line 70 has been plotted through the minimum delay values. The intercept of the line 70 on the minimum delay axis gives the overall propagation delay between the test station and the remote station 12 (that is, the delay that a zero length packet would experience without queuing). This is so because provided that a sufficient number of test packets of each size are submitted, it may be expected that at least one of these packets will pass through the network without queuing and it will be this packet which records the minimum round-trip time.

The FIG. 4 graph can also be used to obtain an estimate of the bandwidth between the test station and the remote station 12 by considering the minimum delay line 70 to be a straight line satisfying the general equation:

$$\text{Overall delay} = (\text{Pacet size}/\text{Bandwidth}) + \text{Propagation Delay}$$

In this case, the bandwidth is the reciprocal of the slope of line 70. In fact, of course, there will generally be a plurality of network hops between the test station and remote station 12 each with its own bandwidth and in these circumstances taking the reciprocal of the slope of the line 70 as the bandwidth is equivalent to saying that:

$$(1/\text{Bandwidth}) = (1/b1 + 1/b2 + \dots 1/bn)$$

where b1, b2 . . . bn are the bandwidths of all the hops in the path. This will only produce a bandwidth estimate close to accurate if there is one hop in the path that is significantly slower than all the others. Where this is not the case, the estimate may be somewhat inaccurate (the inaccuracy arises

because no account is taken of the fact that for a multi-hop path, the intermediate nodes will delay onward transmission of a packet until the whole packet has been received).

The graph of mean delay against packet size shown in FIG. 5 reveals further characteristics of the transmission path between the stations 11 and 12. This graph shows a fragmented line 71, 72 drawn between the mean delay points on the graph; in addition the minimum delay line 70 of FIG. 4 is also shown on the FIG. 5 graph. The difference between the mean and minimum round-trip time delays for the various packet sizes provides an indication of the average queuing delay for each size of packet. The discontinuity in the mean delay line 71, 72 results from the fact that the internal packet size of one or more of the network links has been exceeded by the larger of the transmitted test packets, so that the test packets have been segmented and subject to greater delay in queuing devices of the network. Thus, from FIG. 5 it can be deduced that in the transmission path followed by the test packets between the stations 11 and 12, one or more of the network links has an internal packet size greater than 1000 bytes, but less than 1500 bytes.

In addition to the above described network characteristics that can be derived from the first test (that is bandwidth, propagation delay, average queuing delay and internal network packet size) further characteristics can be derived from the event list 51. These further characteristics include packet lost rate and packet duplication rate.

Instead of, or additionally to the graphical displays illustrated in FIGS. 4 and 5, the sub-routine called by the main program 39 for the purpose of analyzing the event list can be arranged to calculate the various network characteristics described above as being derived from the display graphs.

Second Test Method

The second test implemented by the test apparatus embodying the present invention utilizes a test packet sequence of the form illustrated in FIG. 6. In the FIG. 6 test sequence, a plurality of bursts of back-to-back test packets are provided, each burst being isolated from the other bursts by interval T, such that the effect of one burst on the network has subsided before the next burst is sent. As illustrated, the test packets may all be of the same duration, although for the purposes of the second test, this is not necessary (indeed, the test sequence of FIG. 8 used for the third test described below can also be used for the second test, though the length of the test packets varies through each burst).

The second test is carried out by running the main test sequence program 30 in the manner described above for the first test, except that now the test packet sequence list corresponding to FIG. 6 is accessed by the programme and used to generate the appropriate test sequence. Furthermore, analysis of the event list generated by the FIG. 6 test sequence is affected by a sub-routine specific to the second test.

The purpose of the second test is to identify the minimum buffer length in the transmission path between the test station and the selected remote station 12; this is achieved by saturating the buffer such that packets are lost. In order to achieve the desired saturation, the test sequence includes bursts of packets, each burst effectively being one run through of the second test.

In analyzing and displaying the results of the second test, the event list is processed to determine the mean loss rate of packets for each packet send position in each test sequence burst. The results of this processing are then displayed on the

display 22 in the form of a graph plotting percentage of packets received against send position in burst, as illustrated in the FIG. 7.

In FIG. 7, packets in the first three positions in a burst are shown as being received at a near 100% rate; however, packets in the fourth and fifth position in a burst are only rarely received. A graph of this form provides a strong indication that there is a buffer in the transmission path between the stations 11 and 12 which can only hold three packets before over-flowing. It will be appreciated that not only can this information be derived graphically from the results of analyzing the event list, but it would also be possible to process the analyzed results to derive such an indication.

Third Test Method

The third test implemented by the test apparatus according to the invention is intended to ascertain whether any re-sequencing of packets takes place in transmission between the stations 11 and 12. To this end a test sequence of a form illustrated in FIG. 8 is used. As can be seen, the FIG. 8 test sequence comprises a plurality of test packet bursts isolated from each other by a time interval T. Within a burst, the packets are of decreasing size and are sent back-to-back. A test sequence of this form is likely to identify any re-sequencing behaviour of the network because such behaviour normally involves giving priority to shorter packets and these packets are included towards the end of each burst; if re-sequencing occurs, then it may be expected that the shorter packets will be received earlier in each burst.

The third test is carried out by running the main test sequence program 30 in the manner described above for the first and second test, the primary difference being the program now utilizes a test packet sequence list corresponding to FIG. 8 to control the transmission of test packets. Furthermore, the sub-routine for analyzing and displaying the results of the event list generated by the third test is specific to the third test.

This analysis and display sub-routine is arranged to derive for each receive position in a burst, the mean percentage of packets received by their origin in the transmitted burst. The analysis results are displayed on display 22 in a graph of the form illustrated in FIG. 9. Thus, it can be seen that the first position in a burst received back by the test station 11 is primarily occupied by the first test packet sent out in a burst, but that in addition, the first receive position is also occupied by a small percentage of second and third packets sent out in a burst. It may be deduced from the FIG. 9 graph that re-sequencing of packets does take place over the network path between stations 11 and 12 and that this re-sequencing effects packets at least as large as 1500 bytes (the size of the second test packet in each burst).

Variants

Various modifications and changes to the described test apparatus and methods are, of course, possible. Thus, for example, instead of arranging for each test packet to be looped back by the remote station, the same information regarding network characteristics as derived above using looped back test packets can be derived by one way transmission of the test packet sequences with the remote station collecting each test packet and noting its time of receipt. Such an arrangement is illustrated by the collect and return station 13 in FIG. 1. Because analysis of the test results

requires, at least for some of the tests, a comparison of the time of transmission and of receipt of test packets, the time stamping effected by the collect and return station 13 must be co-ordinated with the transmission time stamping effected by the test station 11. This can be achieved in a number of ways, for example, by arranging for each station to run the Network Time Protocol which serves to synchronize clocks at the stations with a master clock system. After the collect and return station 13 has generated a receive event list similar to the event list 51 of FIG. 2, this list can be returned at the completion of the test sequence to the test station 11 for analysis with the event list generated at the test station at the time of packet transmission.

In fact, it is generally not necessary to have the test and target station clocks closely synchronized with each other. The reason for this is that much of the desired information can be derived by looking at variations in trip time, rather than the actual value of trip time. Trip time variations can be easily determined by knowing the time of transmission from the test station 11 and by having a constant rate clock at the collect and return station 13.

Because the provision of a loop back facility at a network station is very common, carrying out the above described tests using packet loop back is preferred over the collect and return approach embodied by station 13, since in the latter case, special software will need to be provided at the station it is wished to equip with this facility.

Certain network protocols such as the IP protocol with its associated ICMP protocol, provide a loop facility with time stamping at the remote station as well as at the originating station (for IP networks, this facility is provided by ICMP Timestamp messages). In this case, it is possible to derive information on each direction of transit between test and target stations, This generally involves an estimate being made regarding clock offset between the test and target station clocks, this offset being estimated assuming a symmetrical path between the two machines; if this assumption is not approximately correct, this will normally show up as differing values for bandwidth in the two directions.

We claim:

1. A method of testing a packet-based network to ascertain characteristics of packet transmission between first and second nodes on the network, said method comprising the steps of transmitting packets between said first and second nodes; receiving the packets at one of said nodes; and correlating packet transmission and reception to derive correlation data indicative of a first of said transmission characteristics; transmitting said packets as a sequence in which the packets are in a predetermined relationship to each other with at least one parameter of the packets in said sequence varying over the sequence; deriving said correlation data so as to relate the correlation data to one parameter of said at least one parameter; and determining a second of said characteristics of transmission by examination of the relation between the correlation data and said parameter over said sequence.

2. A method according to claim 1, wherein said at least one parameter comprises at least one of inter-packet spacing, packet size and packet position sequence.

3. A method according to claim 2, wherein said at least one parameter is packet size, said sequence of packets comprising for each of a multiplicity of packet sizes, a respective plurality of packets, the packets of said sequence being transmitted in isolation from each other.

4. A method according to claim 3, wherein said correlation data comprises the minimum packet journey time for each packet size, said minimum journey time being graphi-

11

cally displayed against packet size whereby to enable the bandwidth and/or propagation delay between said first and second nodes to be determined.

5. A method according to claim 3, wherein said correlation data comprises the minimum packet journey time for each packet size, said method further comprising the step of processing said correlation data to derive the network bandwidth and/or propagation delay between said first and second nodes.

6. A method according to claim 3, wherein said correlation data comprises the mean packet journey time for each packet size, said mean journey times being graphically displayed against packet size to enable the internal network packet size to be determined.

7. A method according to claim 3, wherein said correlation data comprises the mean packet journey time for each packet size, said method further comprising the step of processing said correlation data to derive the internal network packet size.

8. A method according to claim 3, wherein said correlation data comprises the minimum and mean packet journey times for each packet size, said minimum and mean journey times being graphically displayed against packet size whereby to enable the mean queuing delay between said first and second nodes to be determined.

9. A method according to claim 3, wherein said correlation data comprises the minimum and mean packet journey times for each packet size, said method further comprising the step of processing said correlation data to derive the mean queuing time between said first and second nodes.

10. A method according to claim 2, wherein said at least one parameter is inter-packet spacing and packet position said sequence of packets comprising a succession of isolated packet bursts each made up of a plurality of packets transmitted immediately one after another, said correlation data indication for said at least one parameter expressed as packet position within said bursts as transmitted, the average packet loss rate whereby to enable the minimum buffer capacity between said nodes to be determined.

12

11. A method according to claim 2, wherein said at least one parameter is inter-packet spacing and packet position said sequence of packets comprising at least one isolated packet burst made up of a plurality of packets that are transmitted immediately one after another and are of decreasing size, said correlation data indicating any difference in packet sequencing between the said at least one burst as transmitted and as received.

12. A method according to any one of the preceding claims, wherein said packets are transmitted from said first node and looped-back at said second node for reception at said first node, said correlation data being derived by comparing packet transmission and reception at said first node.

13. A method according to any of claims 1–11, wherein said packets are transmitted from said first node and received at said second node where reception data is derived indicative of parameters of packet reception at said second node, said reception data being transmitted back to said first node to be used as, or in deriving, said correlation data.

14. Apparatus for testing a packet-based network to ascertain characteristics of packet transmission between first and second nodes on the network, said apparatus comprising transmission means for transmitting packets between said first and second nodes, receiving means for receiving the packets of one of said nodes, and correlation means for correlating packet transmission and reception to derive correlation data indicative of a first of said packet transmission characteristics; said transmission means being operative to transmit said packets as a sequence in which the packets are in a predetermined relationship to each other with at least one parameter of the packets in said sequence varying over the sequence; said correlation means being operative to derive said correlation data so as to relate it to one parameter of said at least one parameter; and means for determining a second of said packet transmission characteristics of transmission by examination of the relation between the correlation data and said one parameter over said sequence.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE
CERTIFICATE OF CORRECTION

PATENT NO.   : 5,477,531

DATED   : December 19, 1995

INVENTOR(S) : Neil Howard McKEE et al.

It is certified that error appears in the above-indentified patent and that said Letters Patent is hereby corrected as shown below:

Column 1, line 26, change "ClCMP) to --(ICMP)--.

Column 4, line 1, change "pan" to --part--.

Column 5, line 6, change "CrX/RX" to --(TX/RX--.

Column 7, line 52, change "Pacet" to --Packet--.

Signed and Sealed this

Twenty-seventh Day of August, 1996

Attest:

BRUCE LEHMAN

Attesting Officer   Commissioner of Patents and Trademarks

United States Patent [19]

Mayes et al.

[11] **Patent Number:** 5,793,763

[45] **Date of Patent:** Aug. 11, 1998

[54] **SECURITY SYSTEM FOR NETWORK ADDRESS TRANSLATION SYSTEMS**

[75] Inventors: **John C. Mayes**, Redwood City, Calif.; **Brantley W. Coile**, Athens, Ga.

[73] Assignee: **Cisco Technology, Inc.**, San Jose, Calif.

[21] Appl. No.: **552,807**

[22] Filed: **Nov. 3, 1995**

[51] Int. Cl.^{6} **H04J 3/24**

[52] U.S. Cl. **370/389; 370/401; 370/466; 395/187.01**

[58] Field of Search 370/389, 351, 370/249, 401, 466; 395/186, 187.01

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,962,532	10/1990	Kasiraj et al.	380/25
5,159,592	10/1992	Perkins	370/401
5,287,103	2/1994	Kasprzyk et al.	340/825.52
5,371,852	12/1994	Attanasio et al.	395/200
5,430,715	7/1995	Corbalis et al.	370/54
5,477,531	12/1995	McKee et al.	370/249
5,513,337	4/1996	Gillespie et al.	395/186
5,550,984	8/1996	Gelb	370/466
5,623,601	4/1997	Vu	395/187.01

OTHER PUBLICATIONS

Internet posting for Test Sites to Beta Test an IP Address Translation product; posted on firewalls mailing list; posting made on or after Oct. 28,1994.

Y. Rechkhter, B., Moskowitz, D. Karrenberg, and G. de Groot. "Address Allocation for Private Internets, " RFC 1597, T.J. Watson Research Center, IBM. Corp., Chrysler Corp., RIPE NCC, Mar. 1994.

K. Egevang and P. Francis. "The IP. Network Address Translator (NAT)." RFC 1631, Cray Communications, NTT, May. 1994.
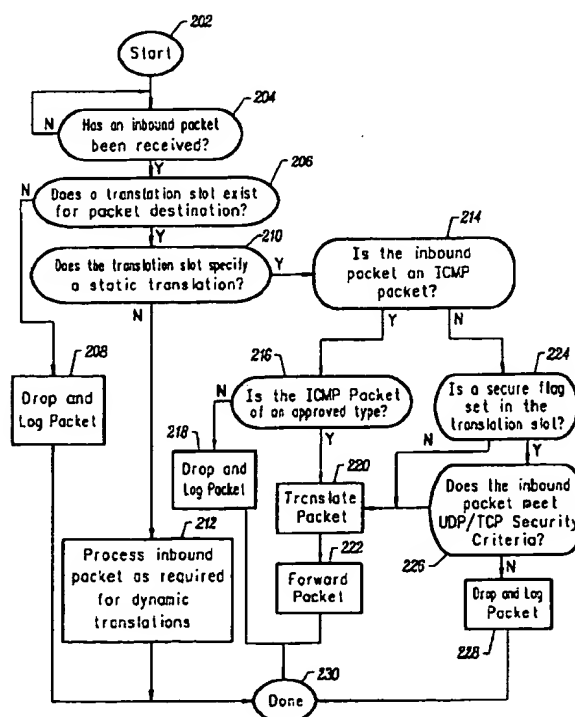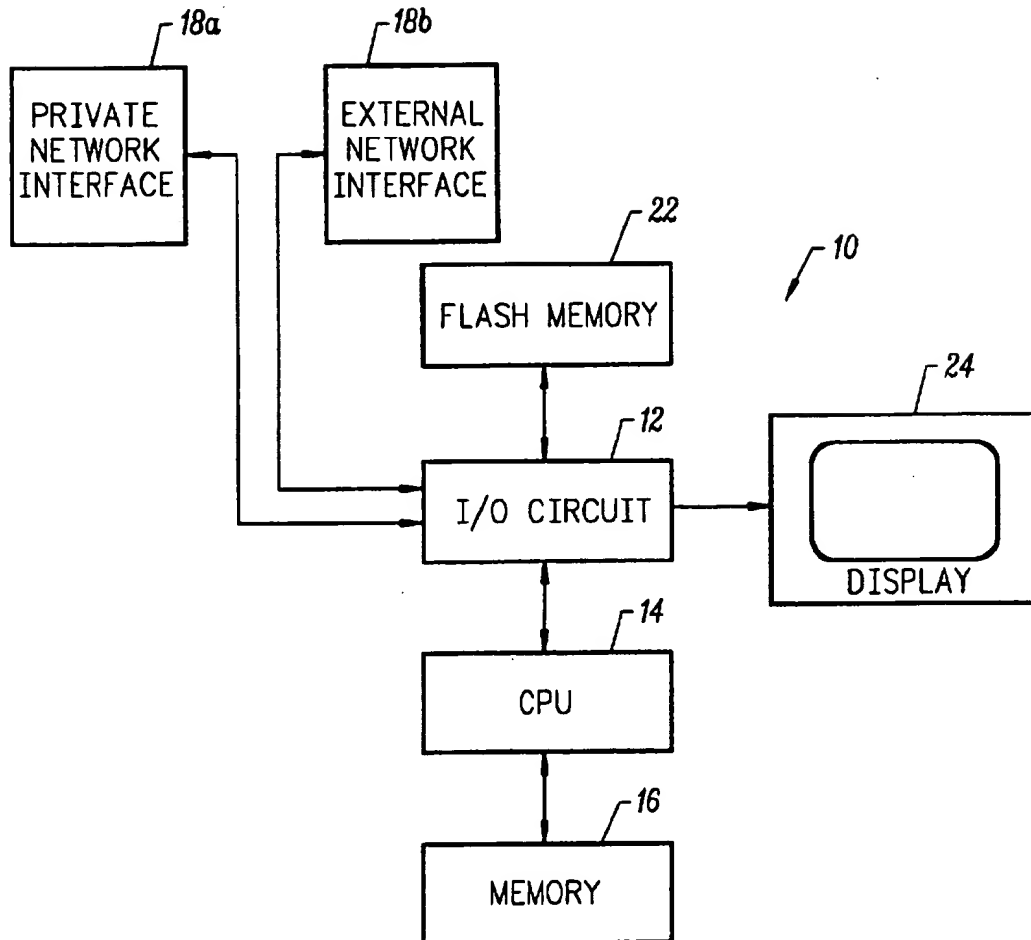
Primary Examiner—Douglas W. Olms
Assistant Examiner—Shick Hom
Attorney, Agent, or Firm—Beyer & Weaver, LLP

[57] **ABSTRACT**

A system and method are provided for translating local IP addresses to globally unique IP addresses. This allows local hosts in an enterprise network to share global IP addresses from a limited pool of such addresses available to the enterprise. The translation is accomplished by replacing the source address in headers on packets destined for the Internet and by replacing destination address in headers on packets entering the local enterprise network from the Internet. Packets arriving from the Internet are screened by an adaptive security algorithm. According to this algorithm, packets are dropped and logged unless they are deemed nonthreatening. DNS packets and certain types of ICMP packets are allowed to enter local network. In addition, FTP data packets are allowed to enter the local network, but only after it has been established that their destination on the local network initiated an FTP session.

42 Claims, 11 Drawing Sheets

FIG. 1

Network Arrangement
32

Internet
38

Enterprise Network
36

Node
56

60b

Router
48

58

64

60b

66

54

62

60a

66

Node
52

(Packet)

50

Translation System

68 (Global IP Address)

60a

70

Pool of IP Addresses

34

(Packet)

42

Internet Destinations

44

60a

46

Router
40
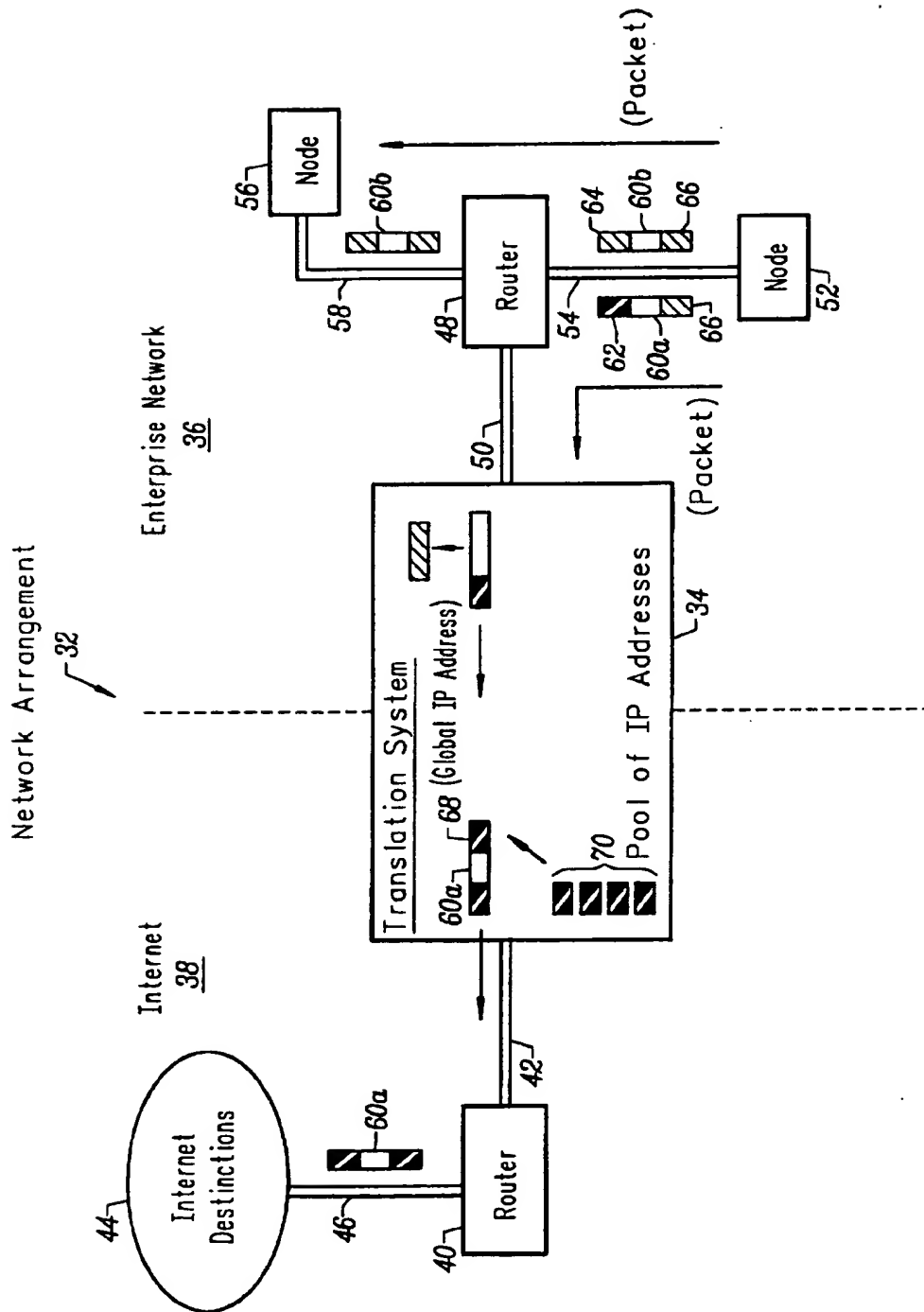
FIG. 2

FIG. 3

Translation Slot

Connection Slot

130 —

Next	—132
Global	—134
Local	—136
Conn	—138
Free	—140
Stamp	—142
Flags	—144
UDP Holes	—146
TCP Holes	—148

FIG. 4A

162 —	Next	— 160
164 —	Flags	
166 —	Faddr	
168 —	F port	
170 —	L port	
172 —	Delta	
174 —	Stamp	
176 —	X fer	
178 —	Start	

FIG. 4B

Start *— 202*

N ← Has an inbound packet been received? *— 204*

Y

N ← Does a translation slot exist for packet destination? *— 206*

Y

Does the translation slot specify a static translation? *— 210*

N

Is the inbound packet an ICMP packet? *— 214*

Y ← (from 210)

Drop and Log Packet *— 208*

Is the ICMP Packet of an approved type? *— 216*

N

Is a secure flag set in the translation slot? *— 224*

N ← (216)

Drop and Log Packet *— 218*

Y

Translate Packet *— 220*

N

Y

Does the inbound packet meet UDP/TCP Security Criteria? *— 226*

Process inbound packet as required for dynamic translations *— 212*

Forward Packet *— 222*

N

Drop and Log Packet *— 228*

Done *— 230*

FIG. 5

226

240
Start

242
Is inbound packet
a UDP packet?

N

246
Is inbound packet
a TCP packet

N

Y

244
Does the UDP packet
meet specified UDP
security criteria?

Y

248
Does the TCP
packet meet specified
security criteria?

N   Y

250
Done
T/F

Y   N

252
Done
D/L

FIG. 6

FIG. 7

FIG. 8

FIG. 9

FIG. 10

FIG. 12

FIG. 11

1

SECURITY SYSTEM FOR NETWORK ADDRESS TRANSLATION SYSTEMS

BACKGROUND OF THE INVENTION

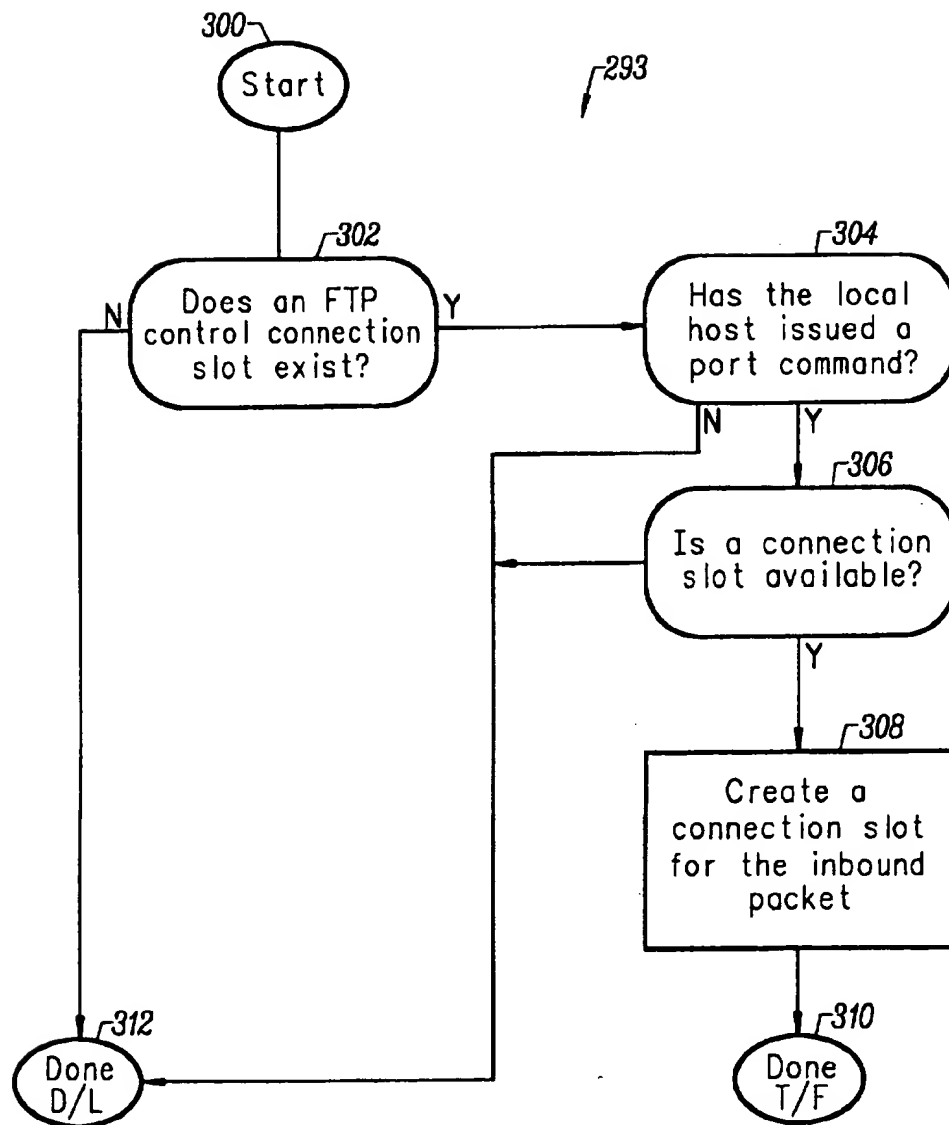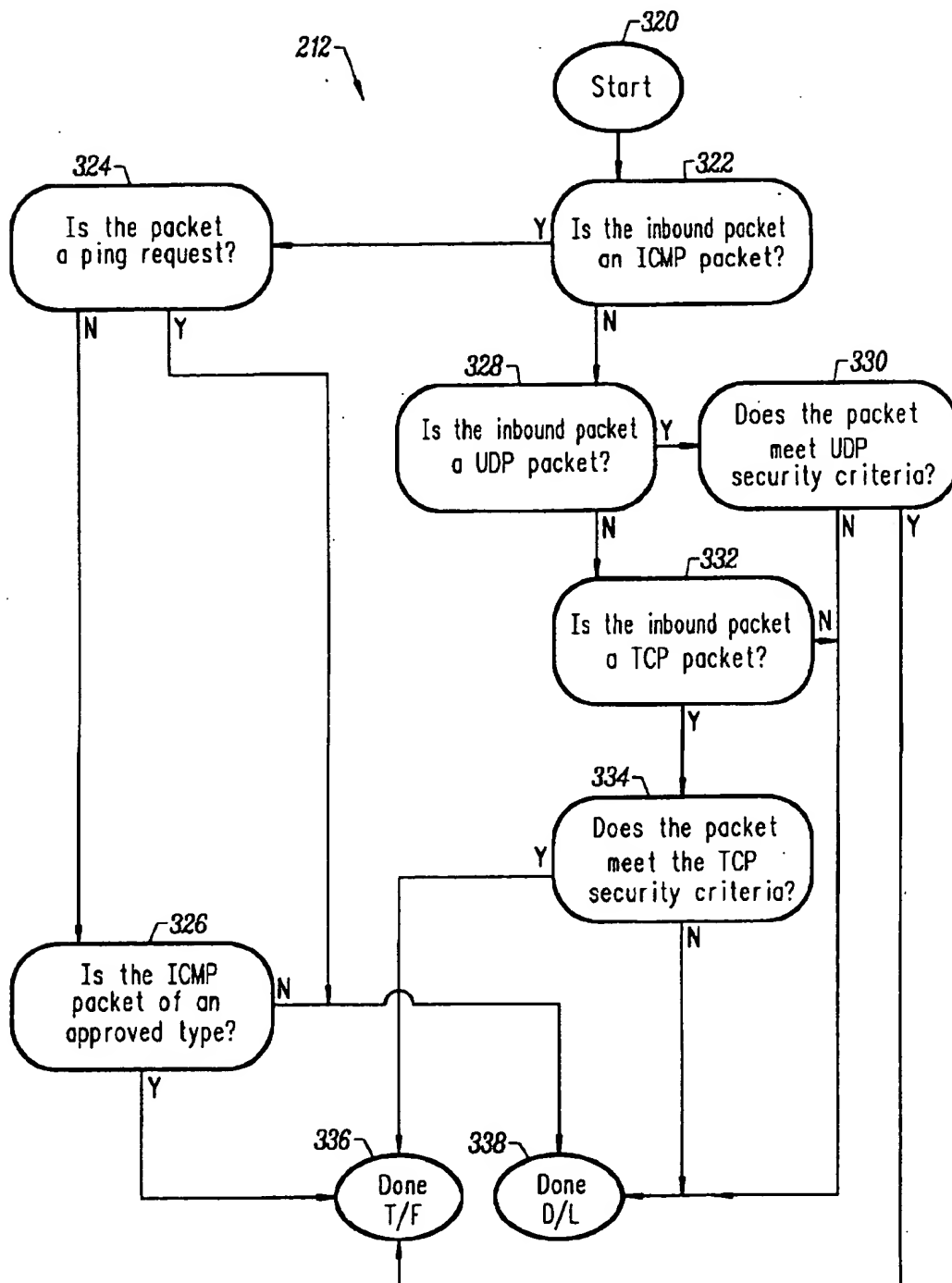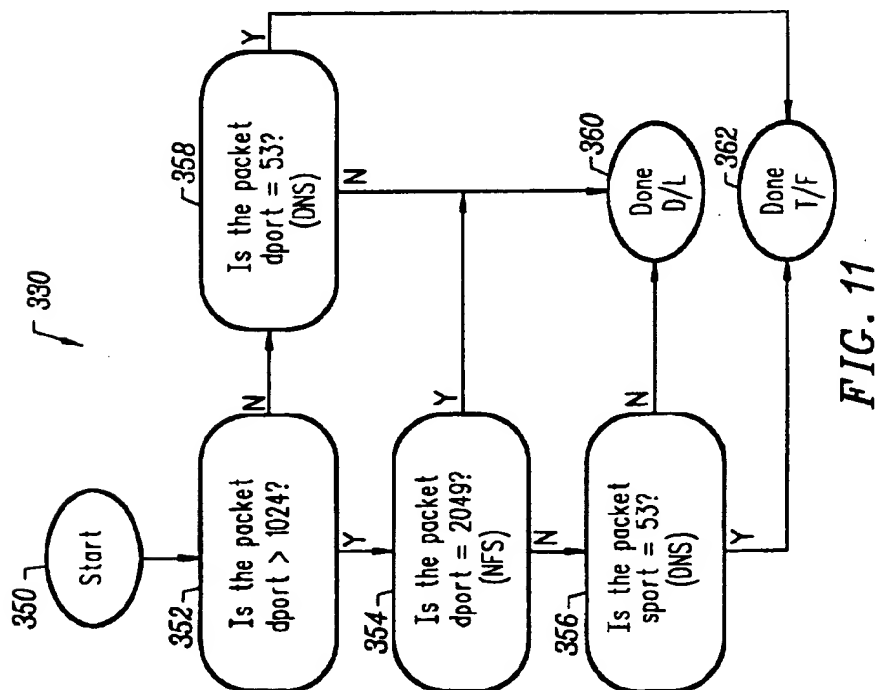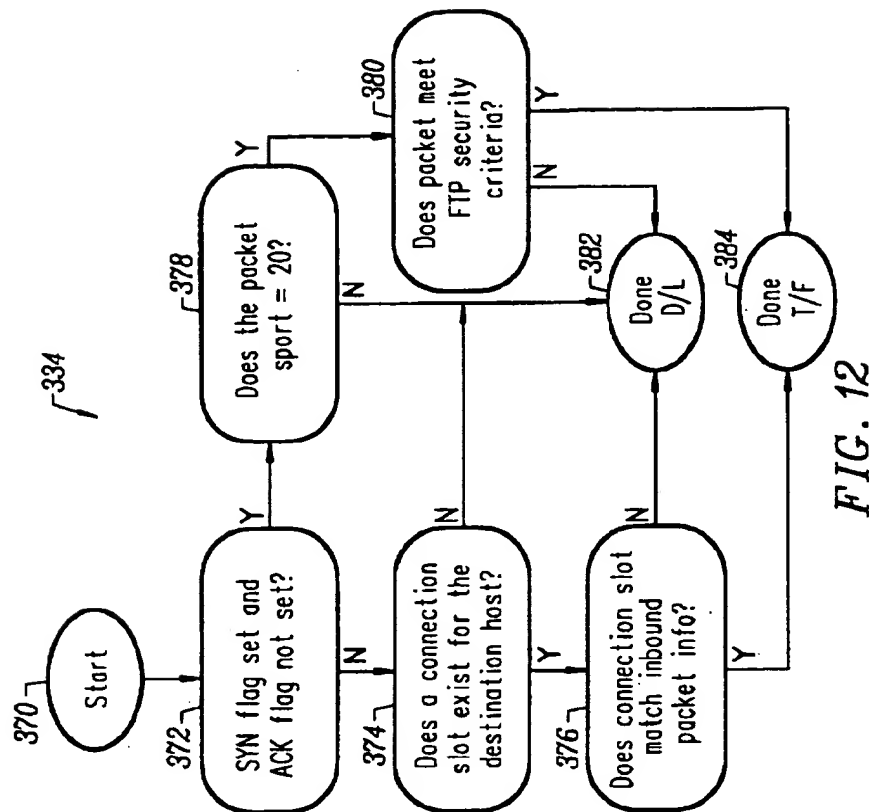The present invention relates to address translation systems for mapping local Internet Protocol "IP" addresses used by hosts on a private network to globally unique IP addresses for communication with hosts on the Internet. The address translation systems have adaptive security mechanisms to protect the private network from certain packet types sent from the Internet.

Private networks are commonly connected to the Internet through one or more routers so that hosts (PCs or other arbitrary network entities) on the private network can communicate with nodes on the Internet. Typically, the host will send packets to locations both within its private network and on the Internet. To receive packets from the Internet, a private network or a host on that network must have a globally unique 32-bit IP address. Each such IP address has a four octet format. Typically, humans communicate IP addresses in a dotted decimal format, with each octet written as a decimal integer separated from other octets by decimal points.

Global IP addresses are issued to enterprises by a central authority known as the Internet Assigned Number Authority ("IANA"). The IANA issues such addresses in one of three commonly used classes. Class A IP addresses employ their first octet as a "netid" and their remaining three octets as a "hostid." The netid identifies the enterprise network and the hostid identifies a particular host on that network. As three octets are available for specifying a host, an enterprise having class A addresses has 2^{24} (nearly 17 million) addresses at its disposal for use with possible hosts. Thus, even the largest companies vastly underuse available class A addresses. Not surprisingly, Class A addresses are issued to only very large entities such as IBM and ATT. Class B addresses employ their first two octets to identify a network (netid) and their second two octets to identify a host (hostid). Thus, an enterprise having class B addresses can use those addresses on approximately 64.000 hosts. Finally, class C addresses employ their first three octets as a netid and their last octet as a hostid. Only 254 host addresses are available to enterprises having a single class C netid.

Unfortunately, there has been such a proliferation of hosts on the Internet, coupled with so many class A and B licenses issued to large entities (who have locked up much address space), that it is now nearly impossible to obtain a class B address. Many organizations now requiring Internet access have far more than 254 hosts—for which unique IP addresses are available with a single class C network address. It is more common for a mid to large size enterprise to have 1000 to 10,000 hosts. Such companies simply can not obtain enough IP addresses for each of their hosts.

To address this problem, a Network Address Translation ("NAT") protocol has been proposed. See K. Egevang and P. Francis, "The IP Network Address Translator (NAT)," Request For Comments "RFC" 1631, Cray Communications, NTT, May 1994 which is incorporated herein by reference for all purposes. NAT is based on the concept of address reuse by private networks, and operates by mapping the reusable IP addresses of the leaf domain to the globally unique ones required for communication with hosts on the Internet. In implementation, a local host wishing to access the Internet receives a temporary IP address from a pool of such addresses available to the enterprise (e.g., class C 254 addresses). While the host is sending and

2

receiving packets on the Internet, it has a global IP address which is unavailable to any other host. After the host disconnects from the Internet, the enterprise takes back its global IP address and makes it available to other hosts wishing to access outside networks.

To implement a NAT, a translation system must be provided between the enterprise private network and the Internet. By virtue of this location, the translation must act as a firewall to protect the local private network from unwanted Internet packets. In view of this requirement, it would be desirable to have a system which employs NAT and provides a secure firewall.

SUMMARY OF THE INVENTION

The present invention provides a system which employs NAT in conjunction with an adaptive security algorithm to keep unwanted packets from external sources out of a private network. According to this algorithm, packets are dropped and logged unless they are deemed nonthreatening. Domain Name Section "DNS" packets and certain types of Internet Control Message Protocol "ICMP" packets are allowed to enter local network. In addition, File Transfer Protocol "FTP" data packets are allowed to enter the local network, but only after it has been established that their destination on the local network initiated an FTP session.

These and other features and advantages of the present invention will be presented in more detail in the following specification of the invention and the figures.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram of a computer system for implementing the processes of a Network Address Translation system in accordance with this invention.

FIG. 2 is a schematic diagram of a private network segment connected to the Internet via a NAT system of this invention.

FIG. 3 is a process flow diagram showing generally the steps involved in transmitting an outbound packet through a NAT system to the Internet in accordance with this invention.

FIG. 4A is a schematic illustration of a translation slot and associated fields in accordance with this invention.

FIG. 4B is a schematic illustration of a connection slot and associated fields in accordance with this invention.

FIG. 5 is a process flow diagram showing generally how an inbound packet is treated by a NAT system of this invention.

FIG. 6 is a process flow diagram illustrating in some detail the security features employed to screen inbound packets destined for a local host having a static translation slot.

FIG. 7 is a process flow diagram depicting a process for screening UDP packets destined for a local host having a static translation slot.

FIG. 8 is a process flow diagram depicting a process for screening TCP packets destined for a local host having a static translation slot.

FIG. 9 is a process flow diagram depicting those steps that may be employed to screen for FTP data destined for a private network.

FIG. 10 is a process flow diagram depicting generally a security algorithm for screening packets destined for a local host having a dynamic translation slot.

FIG. 11 is a process flow diagram depicting a process for screening UDP packets destined for a local host having a dynamic translation slot.

FIG. 12 is a process flow diagram depicting a process for screening TCP packets destined for a local host having a dynamic translation slot.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

1. Definitions

The following terms are used in the instant specification. Their definitions are provided to assist in understanding the preferred embodiments described herein.

A "host" is a PC or other arbitrary network entity residing on a network and capable of communicating with entities outside of its own network through a router or bridge.

A "router" is a piece of hardware which operates at the network layer to direct packets between various nodes of one or more networks. The network layer generally allows pairs of entities in a network to communicate with each other by finding a path through a series of connected nodes.

A "packet" is a collection of data and control information including source and destination node addresses and source and destination ports. The octet of destinations and ports make every connection and packet unique.

2. Overview

The invention employs various process steps involving data manipulation. These steps require physical manipulation of physical quantities. Typically, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated. It is sometimes convenient, principally for reasons of common usage, to refer to these signals as bits, values, variables, characters, data packets, or the like. It should be remembered, however, that all of these and similar terms are to be associated with the appropriate physical quantities and are merely convenient labels applied to these quantities.

Further, the manipulations performed are often referred to in terms, such as translating, running, selecting, specifying, determining, or comparing. In any of the operations described herein that form part of the present invention, these operations are machine operations. Useful machines for performing the operations of the present invention include general purpose and specially designed computers or other similar devices. In all cases, there should be borne in mind the distinction between the method of operations in operating a computer or other processing device and the method of computation itself. The present invention relates to method steps for operating a Network Address Translation system in processing electrical or other physical signals to generate other desired physical signals.

The present invention also relates to an apparatus for performing these operations. This apparatus may be specially constructed for the required purposes, or it may be a general purpose programmable machine selectively activated or reconfigured by a computer program stored in memory. The processes presented herein are not inherently related to any particular computer or other apparatus. In particular, various general purpose machines may be used with programs written in accordance with the teachings herein, or it may be more convenient to construct a more specialized apparatus to perform the required method steps. The general structure for a variety of these machines will appear from the description given below.

Still further, the present invention relates to machine readable media on which are stored program instructions for

performing operations on a computer. Such media includes by way of example magnetic disks, magnetic tape, optically readable media such as CD ROMs, semiconductor memory such as PCMCIA cards, etc. In each case, the medium may take the form of a portable item such as a small disk, diskette, cassette, etc., or it may take the form of a relatively larger or immobile item such as a hard disk drive or RAM provided in a computer.

FIG. 1 shows a typical computer-based system which may be used as a secure Network Address Translation system of the present invention. Shown is a computer 10 which comprises an input/output circuit 12 used to communicate information in appropriately structured form to and from the parts of computer 10 and associated equipment, a central processing unit 14, and a memory 16. These components are those typically found in most general and special purpose computers 10 and are intended to be representative of this broad category of data processors.

Connected to the input/output circuit 12 are inside and outside high speed Local Area Network interfaces 18a and 18b. The inside interface 18a will be connected to a private network, while the outside interface 18b will be connected to an external network such as the Internet. Preferably, each of these interfaces includes (1) a plurality of ports appropriate for communication with the appropriate media, and (2) associated logic, and in some instances (3) memory. The associated logic may control such communications intensive tasks as packet integrity checking and media control and management. The high speed interfaces 18a and 18b are preferably multi-port Ethernet interfaces, but may be other appropriate interfaces such as FDDI interfaces, etc.

The computer system may also include an input device (not shown) such as a keyboard. A flash memory device 22 is coupled to the input/output circuit 12 and provides additional storage capability for the computer 10. The flash memory device 22 may be used to store programs, data and the like and may be replaced with a magnetic storage medium or some other well known device. It will be appreciated that the information retained within the flash memory device 22, may, in appropriate cases, be incorporated in standard fashion into computer 10 as part of the memory 16.

In addition, a display monitor 24 is illustrated which is used to display the images being generated by the present invention. Such a display monitor 24 may take the form of any of several well-known varieties of cathode ray tube displays and flat panel displays or some other type of display.

Although the system shown in FIG. 1 is a preferred computer system of the present invention, the displayed computer architecture is by no means the only architecture on which the present invention can be implemented. For example, other types of interfaces and media could also be used with the computer.

FIG. 2 shows a network arrangement 32 employing a network address translation system 34 of the present invention. Translation system 34 acts as a connection between an enterprise network 36 and the Internet 38. On the Internet side, translation system 34 connects to an Internet router 40 via a line 42. Internet router 40, in turn, connects to Internet destinations 44 through a line 46. On the enterprise network side, translation system 34 connects to a router 48 via a line 50. Router 48 is, in turn, linked to various nodes on the enterprise network 36 including node 52 (via line 54) and node 56 (via line 58).

As an example, assume that node 52 sends packets 60a and 60b to router 48 along line 54. Packet 60a is destined for

the Internet as indicated by a packet header 62. In contrast, packet 60b is destined to for a node on the enterprise network as indicated by packet header 64. Upon receiving packets 60a and 60b, router 48 then routes packet 60b along line 58 to node 56 and routes packet 60a along line 50 to translation system 34.

To this point, the system behaves consistent with most conventional networking protocols. However, packet 60a contains source address 66 which is not a globally unique IP address. Therefore, node 52 can not expect a reply from the Internet destination of packet 60a. To remedy this problem, packet 60a is routed through translation system 34 which modifies the packet so that it can establish a connection with a desired Internet destination. Specifically, when data packet 60a reaches translation system 34, its local source address 66 is replaced with an authorized global IP source address 68 selected from a pool of available global IP addresses 70. Pool 70 includes all or some subset of the global IP source addresses allocated to enterprise network 36.

After packet 60a has been retooled with global IP address 68, translation system 34 sends it along line 42 to Internet router 40. Router 40 then forwards it to the appropriate destination. Thereafter the Internet destination can reply with a packet of its own destined for global IP address 68. Upon receipt of such packet, translation system 34 will determine if it presents a security risk. If not, it will replace address 68 on the inbound packet with the local address of node 52 and then forward the modified packet to router 48. After the node 52 finishes its Internet session, address 68 may be made available to other nodes desiring Internet access. In this manner, a relatively small number of global IP addresses can be used by a much larger network of hosts.

3. Processing of Packets Received by the NAT System

The methods of this invention apply a security algorithm to network address translation. The basic address translation methodolgy may be directly adapted from RFC 1631, previously incorporated by reference.

FIG. 3 details a process 90 that may be employed by network address translation system 34 upon receipt of packet from enterprise network 36. Such outbound packets are received at the inside interface 18a of system 34. The process begins at 94 and in a decision step 96 determines whether an outbound packet has been received from a host on enterprise network 36. If not, the system simply awaits receipt of such packet. If, on the other hand, such packet was indeed received, a decision step 98 determines whether the host sending the packet is listed in a table of allocated translation slots. This table includes a list of global and local IP addresses for all hosts that have a translation slot opened. Translation slots will be described in more detail below. For now, it is sufficient to recognize that a host's local IP address will appear in the table of allocated translation slots if a translation slot has indeed been allocated for that host. To perform step 98, the NAT system first examines the outbound packet source header to identify the local IP address, and then determines if that address is located in the translation slot table. If so, step 98 is answered in the affirmative.

Assuming that step 98 is in fact answered yes (i.e., the translation slot table lists the local IP source address on the packet), a process step 106 examines the actual translation slot for the local host identified in the translation slot table. If on the other hand, step 98 is answered in the negative (i.e., the host sending the packet is not listed in the table of allocated translation slots), a decision step 100 determines

whether a new translation slot is available. If not, an error is logged at process step 102 and the packet is dropped without transmission at a step 104. Thereafter, process control returns to step 96, and system 34 awaits the next outbound packet. Steps 102 and 104 are necessary because the number of translation slots is limited by the number of global IP addresses available to the enterprise network. If the enterprise has only a single class C address collection, for example, no more than 254 translation slots can be used at any given time. The system of this invention does release global IP addresses (i.e., it closes translation slots and removes their entries from the translation slot table) after a defined timeout period. Such period may be adjusted by the network according to the specific network's requirements. A typical default value is 24 hours for example.

Assuming that decision step 100 is answered in the affirmative (i.e., a free translation slot exists), a process step 108 allocates one such translation slot to the host sending the packet. The NAT system the fills the newly allocated slot with various pieces of relevant information (detailed below) including the local host's local IP address and a global IP address selected from the pool of available addresses. In a specific embodiment, the global unique IP address selected from this pool is obtained by simply picking the next available address sequentially. The NAT system also enters the global and local IP addresses for the new translation slot in the translation slot table.

Now, regardless of how a translation slot was identified (via step 106 or 108), the next step is a decision step 110 which determines whether the outbound packet is a Transmission Control Protocol "TCP" packet. As known to those of skill in the art, this determination can be made by checking the appropriate field in the packet header. The TCP protocol requires a connection be established before communication can be commenced.

If the outbound packet turns out not to be a TCP packet, a process step 112 simply translates the IP source address on that packet. In other words, the private source address initially appearing on the packet is replaced with the global unique IP address in the associated translation slot. After the IP source address has been replaced at step 112, a process step 114 fixes the checksums at the end of the packet. Specifically, the address translator will modify the IP checksum and the TCP checksum. Since the differences between the original and translated versions of the packet are known, the checksums are efficiently updated with a simple adjustment rather than a complete recalculation. Details including sample code are provided in RFC 1631. The address translator must also modify those parts of ICMP and FTP packets where the IP address appears. Next, the retooled packet is routed by translation system 34 to the Internet. The process is then complete at 124.

Assuming that decision step 10 determines that the packet is indeed a TCP packet, a decision step 118 then determines whether the "synchronized sequence number" SYN bit has been set in the TCP segment of a TCP header. As known to those skill in the art, this bit is set in the "code bits" section of the TCP header. When the SYN bit is set, it implies that the local host is attempting to establish a connection with a host on the Internet. Assuming that the internal host is in fact attempting to establish a connection, (i.e., decision step 118 is answered in affirmative), translation system 34 creates a new connection slot (if any are available) at a process step 120. That slot is filled information uniquely describing the connection: the remote IP address, the remote port number, and the local port number. Concurrently therewith, the new connection is registered in a "connection field" of the

translation slot. Thereafter, process control is directed to step 112 were the IP source address is translated as described above. Then, the packet checksums are corrected and the packet is routed to the Internet as described above. Assuming that decision step 118 is answered in the negative (i.e., the SYN bit is not set), the system will assume that a TCP session has already been synchronized and locate the connection object associated with internal host's current connection as a step 122. This may be accomplished with a hashing algorithm for example. Thereafter, process control is directed to step 112 where the translation, modification, and forwarding functions are performed as described above. If the outbound packet is a TCP packet without its SYN bit set and no existing connection is open, an error has occurred.

It should be apparent from the above discussion that there is essentially no security mechanism to block outbound packets. Most enterprises expect this behavior.

FIG. 4A is a schematic depiction of a translation slot 130 provided for use with the system/methods of this invention. In practice, the translation slot takes the form of a data structure stored in memory of the NAT system. In the translation slot data structure, a "next" field 132 holds a pointer to the next translation slot in the translation slot table. This field is updated whenever the next successive translation slot times out while the slot at issue remains. A "global" field 134 provides the global unique IP address temporarily held by the host having the translation slot. A "local" address field 136 specifies the local address of the host. The global and local address fields are set when the translation slot is opened and they remain fixed throughout the life of the slot.

A "connection" field 138 contains a listing of the connection slots, if any, appended to the translation slot. More than one connection slot may be associated with a given translation slot, as many users may be using a given host to access the Internet. Each associated process will have its own connection slot. The connection field 138 is updated each time a new connection slot is opened or timed out. Next, a "free" field 140 is reserved for a connection slot of a static translation slot. A "stamp" field 142 provides a time stamp indicating when the translation slot last sent or received a packet. Thus, the stamp field is updated each time an Internet packet passes from or to the local host. This is used for purposes of timing out a translation slot.

Next, a "flags" field 144 contains one or more flags that may be set in connection with the translation slot 130. Examples of such flags include a "static flag" to be set when the translation slot is a "static" rather "dynamic" translation slot. This distinction will be discussed in more detail below. Another flag is a "port" flag to be set when a port command is issued by a local host initiating an FTP session. The User Detection Protocol Holes "UDP Holes" field 146 and the "TCP Holes" field 148 specify "conduits" or exceptions to the adaptive security algorithm of this invention. These conduits which apply only to static translation slots will be discussed in more detail below. The UDP Holes and TCP Holes fields are set by the system administrator when configuring the system with static translation slots.

FIG. 4B is a schematic depiction of a connection slot 160 which may be appended to a translation slot as described above. In this slot, a "next" field 162 holds a pointer to the next connection slot associated with the appropriate translation slot. Next, a "flags" field 164 contains any flags associated with the connection slot. A "faddr" field 166 specifies an address for the foreign host to which the connection is made. A "fport" field 168 specifies a port of the

foreign host. As is known to those of skill in the art, a port is a TCP/IP transport protocol used to distinguish among multiple destinations within a given host computer. Some destinations are reserved for standard services such as e-mail. Next, an "lport" field 170 specifies a port number for the local host. Values are provided to fields 166, 168, and 170 when the connection slot is opened and these fields remain unchanged throughout the life of the connection slot.

A "delta" field 172 specifies an adjustment (delta) to the TCP sequence number as required for address translation in FTP connections. This is because FTP PORT command arguments include an IP address in ASCII. Substituting the ASCII IP address may change the packet size. The delta value specifies by how much the sequence number must be adjusted to account for any difference in packet size due to substitution of the ASCII number. Field 172 must be updated everytime a PORT command is issued. Next, a "stamp" field 174 specifies the time that the connection was last used. This is used for timing out a connection slot. An "xfer" field 176 specifies the total number of bytes transferred while the connection slot is open. The value in this field will continue to grow as more packets are sent and received over the connection. Finally, a "start" field 178 specifies the time that the connection was created.

The process by which translation system 34 handles inbound packets from the Internet (and arriving at NAT system outside interface 18b) is depicted in a process flow diagram 200 shown in FIG. 5. It should be understood that this procedure includes an adaptive security algorithm that does not block outbound packets. In a preferred embodiment, adaptive security follows these rules:

1. Allow any TCP connections that originate from the inside network.

2. Ensure that if an FTP data connection is initiated to a translation slot, there is already an FTP control connection between that translation slot and the remote host. Also ensure that a port command has been issued between the same two hosts. If these criteria are not met, the attempt to initiate an FTP data connection is dropped and logged.

3. Prevent the initiation of a TCP connection to a translation slot from the outside. The offending packet is dropped and logged.

4. Allow inbound UDP packets only from DNS. NFS is explicitly denied.

5. Drop and log source routed IP packets sent to any translation slot on the translation system.

6. Allow only ICMP of types 0, 3, 4, 8, 11, 12, 17 and 18.

7. Ping requests from outside to dynamic translation slots are silently dropped.

8. Ping requests to static translation slots are answered.

Process 200 begins at 202 and then a decision step 204 determines whether an inbound packet has been received. If not, the system simply awaits receipt of such packet. When such packet is received, a decision step 206 determines whether a translation slot exists for the global IP destination address on the packet. If no such translation slot exists, it is impossible to discern which local host is the intended recipient of the packet. Thus, a process step 208 drops and logs the packet. This means that the inbound packet never reaches the enterprise network 36 and its content is logged for post-mortum evaluation by a network administrator.

Assuming that decision step 206 is answered in the affirmative (i.e., a translation slot exists for the incoming packet), a decision step 210 determines whether that translation slot references a static translation. A static translation

slot "hard-wires" to a given internal host machine a globally unique IP address that does not time out. Thus, the host machine maintains an ongoing mapping of its local address to a specific global IP address held by the enterprise. This may be appropriate for internal hosts acting as Internets servers (email, anonymous FTP, World-Wide Web etc.). For all practical purposes, the host machine having the static connection slot appears to be a stand-alone "wide open" node on the Internet. As explained in more detail below, static translation slots unlike dynamic translation slots can have conduits or exceptions to the adaptive security algorithm outlined below.

If decision step 210 is answered in the negative, it can be assumed the translation slot associated with the inbound packet is a dynamic translation slot. In that case, a process step 212 will handle the inbound packet according to a specific algorithm for dynamic translations (see FIG. 10 as discussed below). Thereafter, the process is completed at 230.

Assuming that decision step 210 is answered in the affirmative (i.e., the translation slot is static translation slot), a decision step 214 determines whether the inbound packet is an ICMP frame. As known to those of skill in the art, ICMP packets are used, for among purposes, to handle error and control messages. Many ICMP functions do not pose a security danger. However, others such as a "redirect" (which changes a route in a routing table) pose potential security risks. Thus, if the inbound packet is an ICMP packet a decision step 216 determines whether that packet is of an approved type. Assuming that the ICMP packet is not of an approved type, a process step 218 drops and logs the packet as described above (in the context of step 208). Thereafter, the process is completed at 230.

Assuming that decision step 216 determines that the ICMP packet is of an approved type, a process step 220 translates the inbound packet by replacing the global IP address with a local IP address provided in the static translation slot for the local host receiving the inbound packet. In addition, the system will fix the checksums in the packet as appropriate based upon the address change. Thereafter, at a step 222, the translated inbound packet is forwarded to the local host and the process is completed at 230.

If decision step 214 determines that the inbound packet is not an ICMP packet, a decision step 224 determines whether a "secure flag" is set for the static translation. This can be determined by simply looking a the appropriate field in the static translation slot (field 144) associated with the destination host. Enterprise hosts having static translation slots may or may not employ the adaptive security mechanisms of the present invention. It is up to the user or administrator to configure such hosts appropriately (by setting the secure flag if adaptive security is to be employed). Assuming that decision step 224 is answered in the negative (i.e., the secure flag is not set), process control is directed to step 220 where the inbound packet is translated as described above. Thereafter, the packet is forwarded at step 222 and the process is completed as described above.

If decision step 224 determines that the secure flag is set in the static translation slot, the system will scrutinize the inbound packet according to the adaptive security mechanism at a step 226. Specifically, step 226 will test the inbound packet to determine whether it meets certain "UDP" and "TCP" security criteria. If the step determines that the inbound packet does not pose a security risk, it is translated and forwarded at steps 220 and 222 as described

above. If, however, the inbound packet is found to pose a security risk, it is dropped and logged at a step 228 before the process is completed at 230.

As noted, only certain "nonthreatening" types of ICMP messages will be accepted. Following this paragraph is a list of some ICMP message types. In a preferred embodiment, only types 0, 3, 4, 8, 11, 12, 17 and 18 are allowed. This implies that ICMP redirects (type 5) and others are denied by the adaptive security mechanism.

Type Field	ICMP Message Type
0	Echo Reply
3	Destination Unreachable
4	Source Quench
5	Redirect (change a route)
8	Echo Request
11	Time Exceeded for a Datagram
12	Parameter Problem on a Datagram
13	Timestamp Request
14	Timestamp Reply
15	Information Request
16	Information Reply
17	Address Mask Request
18	Address Mask Reply

The process of determining whether an inbound packet meets the UDP and security criteria of this invention (step 226 of FIG. 5) is depicted in FIG. 6. The process begins at 240 and in a decision step 242 determines whether inbound et is a UDP packet. As known to those of skill art, this information can be readily discerned by checking the appropriate field of an IP datagram header. Assuming that system determines that the inbound packet is indeed a UDP packet, a decision step 244 determines whether that packet meets specified UDP security criteria. If so, the process is completed at 250 with the packet being made available for translation for an forwarding as described with reference to FIG. 5. If, however, decision step 244 determines that the UDP packet does not met the required security criteria, the process is completed at 252 with the UDP packet being dropped and logged as described with respect to step 228 of FIG. 5.

Assuming that decision step 242 determines that the datagram structure of the inbound packet is not a UDP packet, a decision step 246 determines whether that datagram is a TCP packet. If so, a decision step 248 determines whether TCP packet meets the TCP security criteria provide for the translation system. If so, the process is completed at 250 as described above. If not, the process is completed at 252 as described above. Finally, if decision step 246 is answered in the negative (i.e., the inbound packet is neither a UDP nor a TCP packet), the process is completed at 252 as described above.

The process of the determining whether an inbound UDP packet meets the specified UDP security criteria (step 244 of FIG. 6) is detailed in FIG. 7. The process begins a 260, and in a step 262 determines whether any "conduits" exists in the adaptive security mechanism associated with the static translation slot of interest. As described in more detail below, conduits are exceptions to the adaptive security rules described herein Assuming that no such conduits exists, a decision step 264 determines whether the destination port number is greater than 1024. This implies that the destination is a "nonprivileged" port, and there is a probability that the inbound packet is a normal return packet for UDP. Specifically, the packet may be a reply to a DNS (domain name service) request by a local host. If decision step 264 is

answered in the affirmative, a decision step 266 determines whether the destination port is equal to value of 2049. If so, this implies that the inbound packet is an NFS (network file system) packet which should not be accepted. As is known to those of skill in the art, NFS packets are employed to access an external computer's file system as if it was local. Such access from the Internet is clearly inconsistent with the security goals of this invention. Therefore the process is completed at 274 with the packet being dropped and logged pursuant to step 228 of FIG. 5. If, however, decision step 266 is answered in the negative (i.e. the destination is not equal to 2049), a decision step 268 determines whether the source port value is equal to 53. This implies that the inbound packet is a DNS packet which should be accepted so that the local host can access a remote host by name. Thus, if decision step 268 is answered in the affirmative, the process is completed at 272 with the packet being translated in accordance with step 220 of FIG. 5.

The immediately preceding steps allow only those packets having a large destination port (>1024) and a source port equal to 53. This is consistent with DNS requests initiated by the local host.

Assuming that decision step 264 determines that the destination port value is not greater than 1024, a decision step 270 determines whether the destination port value is equal to 53. If so, the inbound packet must be a DNS request packet and should be allowed in. If decision step 270 is answered in the affirmative, the process is completed at 272 as described above. If, on the other hand, decision step 270 is answered in the negative, the process is completed at 274 as described above. In summary, the function of decision steps 264 through 270 is to ensure that the only packets allowed to cross from the Internet through translation system 34 to enterprise 36 are DNS packets. NFS packets are explicitly excluded.

If decision step 262 determines that one or more conduits exists for the static translation slot, a decision step 263 determines whether the UDP packet matches any of those conduits. If so, the process is completed at 272 as described above. If not, process control is directed to step 264 for evaluation of the packet's destination port as described above.

As mentioned, conduits are exceptions to the general adaptive security rules implemented on translation system 34. Such rules were summarized above. Each secure static translation slot will have zero or more conduits provided therewith in the "UDP Holes" and/or "TCP Holes" fields 146 and 148 (see FIG. 4A). These fields are supplied with specific conduits when the translation system is configured. Each such conduit is provided in the format protocol: address/mask-port. The "protocol" section specifies either UDP or TCP, the "address" section specifies a destination global IP address, the net "mask" section specifies how many bits of the IP address that are to be compared in the matching step, and the "port" section specifies a destination port value (e.g., 25 for mail, 24 for Telnet, etc.). The "mask" section can be set as large or small as desired. For example, if an administrator wishes to allow an entire class C address group into the conduit, the mask would be set to compare only the 24 IP address bits specifying netid. If the information contained in the inbound packet meets the criteria specified by a conduit, the conduit is "matched" pursuant to step 263 of FIG. 7.

Multiple exceptions may be applied to a single static translation slot (via multiple conduit commands). This allows the administrator to permit access from an arbitrary

machine, network, or any host on the Internet to the inside host defined by the static translation slot.

FIG. 8 details the step of determining whether a TCP packet meets the specified security criteria (step 248 of FIG. 6). The process begins at 280 and in a decision step 282 determines whether a conduit exists for the static translation slot at issue. If so, a decision step 284 determines whether the information contained in the inbound packet matches any conduit for the static translations slot. This process is conducted exactly as described with reference to decision step 263 in FIG. 7. If decision step determines that there is a conduit match, the process is completed at 294 with translation and forwarding of the packet. If however, decision step 284 determines that there is no conduit match, a decision step 286 determines whether the SYN flag is set and the ACK flag is not set. If so, an external source is likely attempting an unsolicited connection to the enterprise network. In general, such packets should not be accepted unless they are part of a requested FTP file transfer.

If decision step 282 determines that there are no conduits associated with the static translation slot of the destination host, process control is directed to decision step 286. If the conditions of step 286 are not met (i.e., either the SYN flag is not set or the ACK flag is set), a decision step 288 determines whether a connection slot exists for the static translation slot at issue. If no such connection slot exists, it can be assumed that no connection was initiated by a host in the enterprise network 36. Thus, if decision step 288 is answered in the negative, the process is concluded at 296 with the packet being dropped and logged. If, however, a connection slot does exist for the static translation slot, a decision step 290 determines whether the port and IP source and destination addresses of the inbound packet match those of any connection object of the static translation slot. If no such match is found, it can again be assumed that the internal host did not initiate a connection requesting the inbound packet. Thus, if decision step 290 is answered in the negative, the process is concluded at 296 with the packet being dropped and logged. If, on the other hand, decision step 290 is answered in the affirmative (that is, the port and IP addresses in the inbound packet match those of a connection object), the process is concluded at 294 with the packet being translated and forwarded.

If decision step 286 is answered in the affirmative, a decision step 292 determines whether the source port value of the inbound packet equals 20. A source port value of 20 indicates that an FTP (file transfer protocol) data connection is being established. If decision step 292 is answered in the negative (i.e., the inbound packet is attempting to establish a connection for some purpose other than sending FTP data), the process is concluded at 296 with the packet being dropped and logged. If, however, decision step 292 is answered in the affirmative, a decision step 293 determines whether the inbound packet meets certain FTP security criteria. If so, the process is completed at 294 with the packet being translated and forwarded. If not the process ends at 296 with the packet being dropped and logged.

FIG. 9 details the process steps associated with determining whether FTP security criteria have been met (step 293 of FIG. 8). The process begins at 300 and then a decision step 302 determines whether an FTP control connection slot exists for the translation slot of the local host. This can be determined by a destination port number of 21 in the "fport" field 168 of a connection slot for the local host (see FIG. 4B). If so, the host associated with the static translation slot has initiated an FTP control session and the inbound packet may possibly accepted. As is known to those of skill in the art,

FTP consists of two connections: first a local host logs into a remote server with a control connection and then the remote server responds with a data connection to the local host. Assuming that an FTP control connection slot exists, a decision step 304 determines whether the local host has issued a PORT command. This may be established by checking for a "port" flag in the flags field of the translation slot (see FIG. 4A). Assuming that such port command has been issued, a decision step 306 determines whether a new connection slot is available. It is possible that the translation system 34 may have too many simultaneous connections to allocate a new connection slot. That is, it is of course possible that all available connection slots are in use. If, however, one or more additional connection slots are available (i.e., decision step 306 is answered in the affirmative), a process step 308 creates a new connection slot for the inbound FTP data packet. Concurrently therewith, the new connection is registered in the "connection field" of the translation slot. Thereafter the process is completed at 310 with the inbound packet being translated and forwarded. If any of decision steps 302, 304, or 306 are answered in the negative, the process is concluded at 312 with the packet being dropped and logged.

As noted above in the context of FIG. 5, inbound packets that are not destined for static translation slots may be destined for dynamic translation slots. In fact, all translation slots are either static or dynamic. Thus, if a translation slot exists for the destination of an inbound packet, and that destination does not have a static translation slot, then it must have a dynamic translation slot. As noted in connection with the discussion of FIG. 5, a decision step 212 determines whether such inbound packets meet the security requirements for dynamic translation slots. FIG. 10 details a process by which such security requirements are evaluated. The process begins at 320, and in a decision step 322 the translation system determines whether the inbound packet is an ICMP packet. If so, decision step 324 determines whether that packet contains a "ping" request (ICMP echo message—type 8). If so, the process is completed at 338 with the inbound packet being dropped and logged. If on the other hand, the ICMP packet is not a ping request, a decision step 326 determines whether the inbound packet is one of the approved types of packets. If decision step 326 determines that the inbound ICMP packet is not one of the approved types, the process is concluded at 338 with the packet being dropped and logged. If, on the other hand, the ICMP packet is of an approved type, the process is concluded at 336 with the packet being translated and forwarded.

A comparison of the processes depicted in FIGS. 5 and 10 shows that ICMP packets are treated similarly when destined for hosts with either static or dynamic translation slots. However, a host with a static translation slot will accept a ping request (ICMP message type 8) while a host with a dynamic translation slot will not. Thus, external sources can establish the presence of static hosts but not dynamic hosts. This is because the dynamic hosts change IP addresses from time to time and are intended to be shielded behind translation system.

· Returning now to FIG. 10, assuming that the inbound packet is not an ICMP packet (i.e., decision step 322 is answered in the negative), a decision step 328 determines whether the inbound packet is a UDP packet. If so, a decision step 330 determines whether the inbound UDP packet meets the security criteria appropriate for a dynamic translation slot. If so, the process is concluded at 336 with the packet being translated and forwarded. If not, the process is concluded at 338 with the packet being dropped and logged.

Assuming that the inbound packet is neither an ICMP packet nor a UDP packet (i.e., both decision steps 322 and 328 are answered in the negative), a decision step 332 then determines whether the inbound packet is a TCP packet. If not, the process is concluded at 338 with the packet being dropped and logged. If, on the other hand, the packet is indeed a TCP packet, a decision step 334 determines whether that packet meets the adaptive security criteria required of a TCP packet destined for a host having a dynamic translation slot. If so, the process is concluded at 336 with the packet being translated and forwarded. If not, the process is concluded at 338 with the packet being dropped and logged.

The UDP and TCP security criteria for inbound packets are nearly identical for hosts having static translation slots and hosts having dynamic translation slots. The only difference is that hosts with static translation slots can have conduits or exceptions to the security mechanism. The following discussion of FIGS. 11 and 12 will illustrate this.

The process of the determining whether an inbound UDP packet meets security criteria for a dynamic slot (step 330 of FIG. 10) is detailed in FIG. 11. The process begins a 350, and a decision step 352 evaluates the incoming packet to determine whether its destination port number is greater than 1024. If so, a decision step 354 determines whether the destination port is equal to a value of 2049. If so, this implies that the inbound packet is an NFS (network file system) packet which should not be accepted. In such cases, the process is completed at 360 with the packet being dropped and logged. If, however, decision step 354 is answered in the negative (i.e. the destination is not equal to 2049), a decision step 356 determines whether the source port value is equal to 53. This implies that the inbound packet is a DNS packet which should be accepted. Thus, if decision step 356 is answered in the affirmative, the process is completed at 362 with the packet being translated and forwarded.

Assuming that decision step 352 determines that the destination port value is not greater than 1024, a decision step 358 determines whether the destination port value is equal to 53. If so, the inbound packet must be a DNS request packet and should be allowed in. Thus, if decision step 358 is answered in the affirmative, the process is completed at 362 as described above. If, on the other hand, decision step 358 is answered in the negative, the process is completed at 360 as described above. As in the case of static translation slots, the only packets allowed to cross from the Internet through translation system 34 to enterprise network 36 are DNS packets. NFS packets are explicitly excluded.

FIG. 12 details the process of determining whether a TCP packet meets the security criteria for a dynamic slot (step 334 of FIG. 10). The process begins at 370 and in a decision step 372 examines the incoming packet and determines whether the SYN flag is set and the ACK flag is not set. If not, a decision step 374 determines whether a connection slot exists for the dynamic translation slot at issue. If no such connection slot exists, it can be assumed that no connection was initiated by a host in the enterprise network 36. Thus, if decision step 374 is answered in the negative, the process is concluded at 382 with the packet being dropped and logged. If, however, a connection slot does exist for the dynamic translation slot, a decision step 376 determines whether the port and IP source and destination addresses of the inbound packet match those of any connection object of the dynamic translation slot. If no such match is found, it can again be assumed that the internal host did not initiate a connection requesting the inbound packet. Thus, if decision step 376 is answered in the negative, the process is concluded at 382

15

with the packet being dropped and logged. If, on the other hand, decision step 376 is answered in the affirmative (that is, the port and IP addresses in the inbound packet match those of a connection object), the process is concluded at 384 with the packet being translated and forwarded.

If decision step 372 is answered in the affirmative, a decision step 378 determines whether the source port value of the inbound packet equals 20. As noted, a source port value of 20 indicates that an FTP data connection is being established. If decision step 378 is answered in the negative (i.e., the inbound packet is attempting to establish a connection for some purpose other than sending FTP data), the process is concluded at 382 with the packet being dropped and logged. If, however, decision step 378 is answered in the affirmative, a decision step 380 determines whether the inbound packet meets certain FTP security criteria. If so, the process is completed at 384 with the packet being translated and forwarded. If not the process ends at 382 with the packet being dropped and logged. The FTP security criteria referenced in step 380 may be identical to those set forth in FIG. 9.

Although the foregoing invention has been described in some detail for purposes of clarity of understanding, it will be apparent that certain changes and modifications may be practiced within the scope of the appended claims. For example, the private network described above may be a single local area network or multiple local area networks connected as a wide area network. Further, the adaptive security algorithm described above may be applied to a single machine as well as a network.

What is claimed is:

1. A method for translating network addresses on packets destined for local hosts on a private network from hosts on an external network, the method comprising the following steps:

identifying a global IP destination address on an inbound packet arriving at the private network;

determining whether the global IP destination address corresponds to any local host on the private network by determining if a translation slot data structure exists for the global IP destination address, which translation slot associates the global IP destination address to a corresponding local IP address for a particular local host which has sent an outbound packet to an external network host on the external network within a defined time period;

if the inbound packet is found to be intended for the particular local host on the private network which has sent the outbound packet to the external network host within said defined time period, determining whether the inbound packet meets defined security criteria;

if the inbound packet meets said security criteria, replacing the inbound packet's global IP destination address with the corresponding local IP address for the particular local host to which the inbound packet was addressed; and

forwarding the inbound packet to the particular local host to which the inbound packet was addressed.

2. The method of claim 1, where in said global IP address is one of a collection of addresses shared by hosts on said private network.

3. The method of claim 2, wherein there are fewer global IP addresses in the collection of IP addresses than there are hosts on said private network.

4. The method of claim 1, further comprising the following steps:

16

creating the translation slot data structure when the particular local host on the private network sends an outbound packet to an external network destination;

removing the translation slot data structure after said defined time period has elapsed.

5. The method of claim 1, further comprising a step of determining whether the global IP destination address is statically fixed to the particular local host on the private network; and, if so, employing a first set of defined security criteria.

6. The method of claim 5, further comprising a step of employing a second set of defined security criteria if the global IP destination address is not statically fixed to the particular local host.

7. The method of claim 1, wherein said step of determining whether the inbound packet meets defined security criteria includes a step of determining whether the inbound packet is a ping request and, if so, dropping the inbound packet.

8. The method of claim 1, wherein said step of determining whether the inbound packet meets defined security criteria includes a step of determining whether the inbound packet is an ICMP packet of an approved type.

9. The method of claim 8, wherein the approved types of ICMP packets include packets of types 0, 3, 4, 8, 11, 12, 17, and 18.

10. The method of claim 1, wherein said step of determining whether the inbound packet meets defined security criteria includes a step of determining whether the inbound packet has a destination port number corresponding to a nonprivileged port of the local host.

11. The method of claim 1, wherein said step of determining whether the inbound packet meets defined security criteria includes a step of determining whether the inbound packet has a source or destination port number corresponding to a domain name service request.

12. The method of claim 1, wherein said step of determining whether the inbound packet meets defined security criteria includes a step of determining whether the inbound packet has a destination port number corresponding to a network file system packet.

13. The method of claim 1, wherein said step of determining whether the inbound packet meets defined security criteria includes a step of determining whether the inbound packet is a TCP packet.

14. The method of claim 13, wherein said step of determining whether the inbound packet meets defined security criteria further includes a step of determining whether the particular local host on the private network initiated a connection requesting the inbound TCP packet.

15. The method of claim 14, wherein said step of determining whether the inbound packet meets defined security criteria further includes the following steps:

determining whether the inbound packet is a file transfer protocol packet; and

if so, determining whether the inbound packet meets FTP security criteria.

16. A network address translation system for translating network addresses on packets sent from an external host on an external network to a local host on a private network, the private network having a plurality of local hosts at least some of which communicate with hosts on the external network, the network address translation system comprising:

an outside interface connected to the external network;

an inside interface connected to the private network; and

a translation slot data structure stored on the network address translation system, the translation slot specify-

ing at least (i) a global IP address temporarily held by the local host, (ii) a local address fixed with local host, wherein the network address translation system creates the translation slot when the local host sends a packet to said external host and times out the translation slot after a defined time period has elapsed.

17. The network address translation system of claim 16, wherein said translation slot data structure includes at least a first field for the global IP address and a second field for the local address.

18. The network address translation system of claim 17, wherein the multiple fields further include one or more holes fields which may be set to specify that exceptions to a security algorithm employed by the network address translation system to prevent suspicious packets from reaching the private network.

19. The network address translation system of claim 16, wherein the network address translation system creates said translation slot only when the local host sends a packet to said external host.

20. The network address translation system of claim 16, wherein the translation slot includes a stamp field specifying a time when the network address translation system created the translation slot and wherein the network address translation system measures said defined period of time from time specified in the stamp field.

21. The network address translation system of claim 16, further comprising a connection slot appended to said translation slot and specifying that a TCP connection has been established between the local host and the external host, wherein the network address translation system creates the connection slot when the local host sends a TCP packet to said external host to initiate the TCP connection, and times out the connection slot after a defined time period has elapsed since the TCP connection was last used.

22. The network address translation system of claim 21, further comprising a memory, wherein the connection slot is a data structure is comprised of multiple fields and stored on the memory, and wherein the multiple fields include at least a first field for the external host, a second field for a TCP/IP port of the external host, and a third field for a TCP/IP port of the local host.

23. A network address translation system for translating network addresses on packets sent from an external network having a plurality of external hosts to a local host on a private network having a plurality of local hosts at least some of which communicate with hosts on the external network, the network address translation system comprising:

an outside interface connected to the external network;
an inside interface connected to the private network;
means for identifying a global IP destination address on an inbound packet arriving at the private network;
means for determining if a translation slot data structure exists for the global IP destination address, which translation slot associates the global IP destination address with a corresponding local IP address of a particular local host on the private network, which particular local host has sent an outbound packet to an external network host within a defined time period;
means for determining whether the inbound packet meets defined security criteria if the inbound packet is found to be addressed to the particular local host on the private network which has sent the outbound packet to the external network host within said defined time period;
means for replacing the inbound packet's global IP destination address with a corresponding local IP address

for the particular local host to which the inbound packet was addressed; and
means for forwarding the inbound packet to the particular local host to which the inbound packet was addressed.

24. The network address translation system of claim 23, further comprising means for selecting said global IP address from among a collection of IP addresses available to hosts on said private network.

25. The network address translation system of claim 24, wherein there are fewer global IP addresses in the collection of IP addresses than there are hosts on said private network.

26. The network address translation system of claim 25, further comprising:
means for creating the translation slot data structure when the particular local host on the private network sends an outbound packet to an external network destination;
means for removing the translation slot data structure after said defined time period has elapsed.

27. The network address translation system of claim 23, further comprising means for determining whether the global IP destination address is statically fixed to the particular local host on the private network, and, if so, employing a first set of defined security criteria.

28. The network address translation system of claim 27, further comprising means for employing a second set of defined security criteria if the global IP destination address is not statically fixed to the particular local host.

29. The network address translation system of claim 23, wherein said means for determining whether the inbound packet meets defined security criteria determines whether the inbound packet is an ICMP packet of an approved type.

30. The network address translation system of claim 29, wherein the approved types of ICMP packets include packets of types 0, 3, 4, 8, 11, 12, 17, and 18.

31. The network address translation system of claim 23, wherein the means for determining whether the inbound packet meets defined security criteria determines whether the inbound packet has a source or destination port number corresponding to a domain name service request.

32. The network address translation system of claim 23, wherein the means for determining whether the inbound packet meets defined security criteria determines whether the inbound packet has a destination port number corresponding to a network file system packet.

33. The network address translation system of claim 23, wherein the means for determining whether the inbound packet meets defined security criteria determines whether the inbound packet is a TCP packet.

34. The network address translation system of claim 33, wherein the means for determining whether the inbound packet meets defined security criteria also determines whether a local host on the private network initiated a connection requesting the inbound TCP packet.

35. The network address translation system of claim 34, wherein the means for determining whether the inbound packet meets defined security criteria determines whether the inbound packet is a file transfer protocol packet, and, if so, determines whether the inbound packet meets FTP security criteria.

36. A method for translating network addresses on packets destined for local hosts on a private network from hosts on an external network, the method comprising:
identifying a first global IP destination address on an inbound packet arriving at the private network, which first global IP address is one of a collection of global IP addresses available to the local hosts on the private network;

19

determining that the first global IP destination address corresponds to a particular local host on the private network by locating translation data specifying the first global IP destination address and associating it with a corresponding local IP address of the particular local host which has sent an outbound packet to an external network host on the external network within a defined time period, which outbound packet has had the local IP address replaced with the first global IP address;

determining whether the inbound packet meets defined security criteria;

if the inbound packet meets said security criteria, forwarding the inbound packet to the particular local host to which the inbound packet was addressed.

37. The method of claim 36, further comprising replacing the first global IP destination address on the inbound packet with the corresponding local IP address prior to forwarding the inbound packet to the particular local host.

38. The method of claim 36, wherein there are fewer global IP addresses in the collection of global IP addresses than there are hosts on said private network.

39. A network address translation system for translating network addresses on packets sent from an external host on an external network to a local host on a private network, the private network having a plurality of local hosts at least

20

some of which communicate with hosts on the external network, the network address translation system comprising:

an outside interface connected to the external network;

an inside interface connected to the private network; and

a memory on which is stored

(a) a collection of global IP addresses available to the local hosts on the private network, and

(b) translation data associating at least (i) a global IP address temporarily held by the local host and (ii) a local IP address fixed with the local host.

40. The network address translation system of claim 39, wherein the system is configured to replace the local IP address with the global IP address on outbound packets from the local host to the external host.

41. The network address translation system of claim 39, wherein the network address translation system creates the translation data only when the local host sends a packet to said external host.

42. The network address translation system of claim 39, wherein the translation data includes a stamp field specifying a time when the network address translation system created the translation data.

* * * * *